بسم الله الرحمن الرحيم

الكتاب العربى لإدارة قاعدة البيانات اوركل

الجزء الاول

Oracle database 10g

Obay Salah Mirgani (Oracle 10*g* DBA OCP)

obaysalah@hotmail.com

الفهرس

الإهداء

إلى رِحـــابك دًبجًنا رسائلنا لهبا ... تكاد تُحــرق من اشواقنا

يا قارئ الحرف اهديناك احرفنا منسكبا ... وقبلها قد بعثنا الدمــع

شوقاً إليك فهل ترضـى محبتنا والهدبا ... مهراً وإلا قد بعثنا القلب

فغيرنا بمداد الحــبر قد كتبوا والخطبا ... ومن دمانا كتبنا الشعر

المقدمة

هذا هو الجزء الاول من الكتاب العربى لإدارة قاعدة البيانات اوركل والذى هو على جزئين ، وحسبى من هذا الكتاب أنى كتبته لى ولامثالى ، فأنا اول المستفدين منه ، فإنى اعود له كل مره وقد خططته بيدى فأجد فيه المعلومة الشاردة التى تعودت أن اصل اليها بعد بحث عميق.

هذا الكتاب يقول لك هكذا تنجز مهامك فى إدارة قاعدة البيانات بأسط الطرق وبأقل جهد ودون تَكلف او تعقيد ، حاولت أن اتناول فيه المادة العلمية بإسلوب سهل بسيط ؛ فكنت احياناً أجمع عدة مواضيع فى فصل واحد لارتباطهما فى الفكرة والمضمون ، واحياناً اقدم واحياناً أُرجئ الى حيث تكون المعلومة افيد واقيم.

أما الجانب العملى فكان له اوًفر الحظ والنصيب ، فقد طرحته فى شكلين ، الاول كتابة النصوص العملية فى قوالب بحيث يستطيع القارئ نسخها وتنفيذها او تعديلها متى شاء ، والثانى صور لتنفيذ النصوص على محرر الSQL*PLUS لتثبت لك الفكرة بأكثر من طرح.

املى أن يستفيد القارئ العربى من هذا الكتاب الذى ناقشت فيه جزء من اهم المواضيع المتعلقة بإدارة قاعدة البيانات اوركل ، وبقىَ الجزء الاخر سوف اطرحه فى الجزء الثانى من الكتاب والذى سوف يكون بين يديك قريباً لو امًد الله لنا فى الاعمار.

وقبل البدء لى عندك طلب ايها القارئ ، دعوة خالصة منك أن يعفو الله عنى فإنى لا أعلم أحد عنده من الذنوب مثل ما لدى ولكن أسأل الله أن يديم علينا ستره فى الدنيا والاخره.

كتبه أبى صلاح ميرغنى (لباشا)

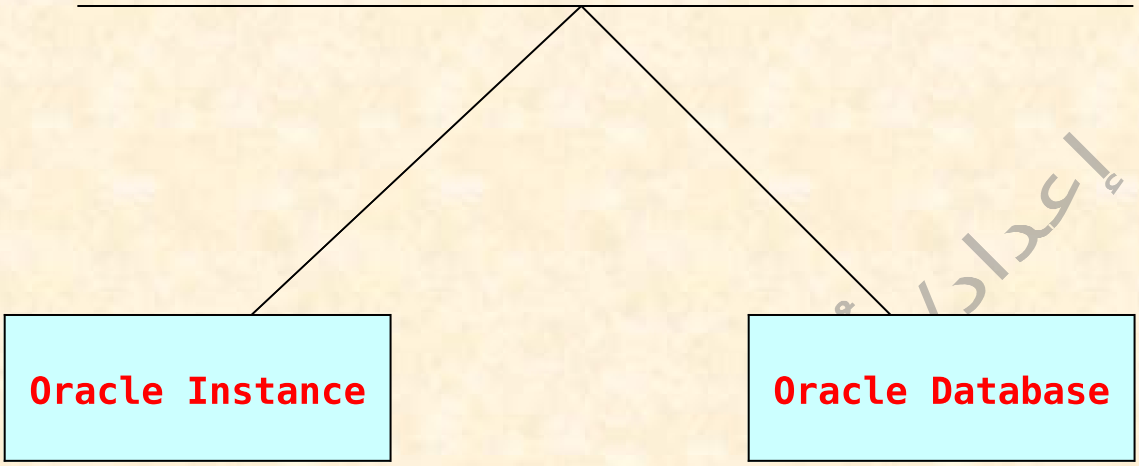
الفصل الاول

مكونات قاعدة البيانات أوركل

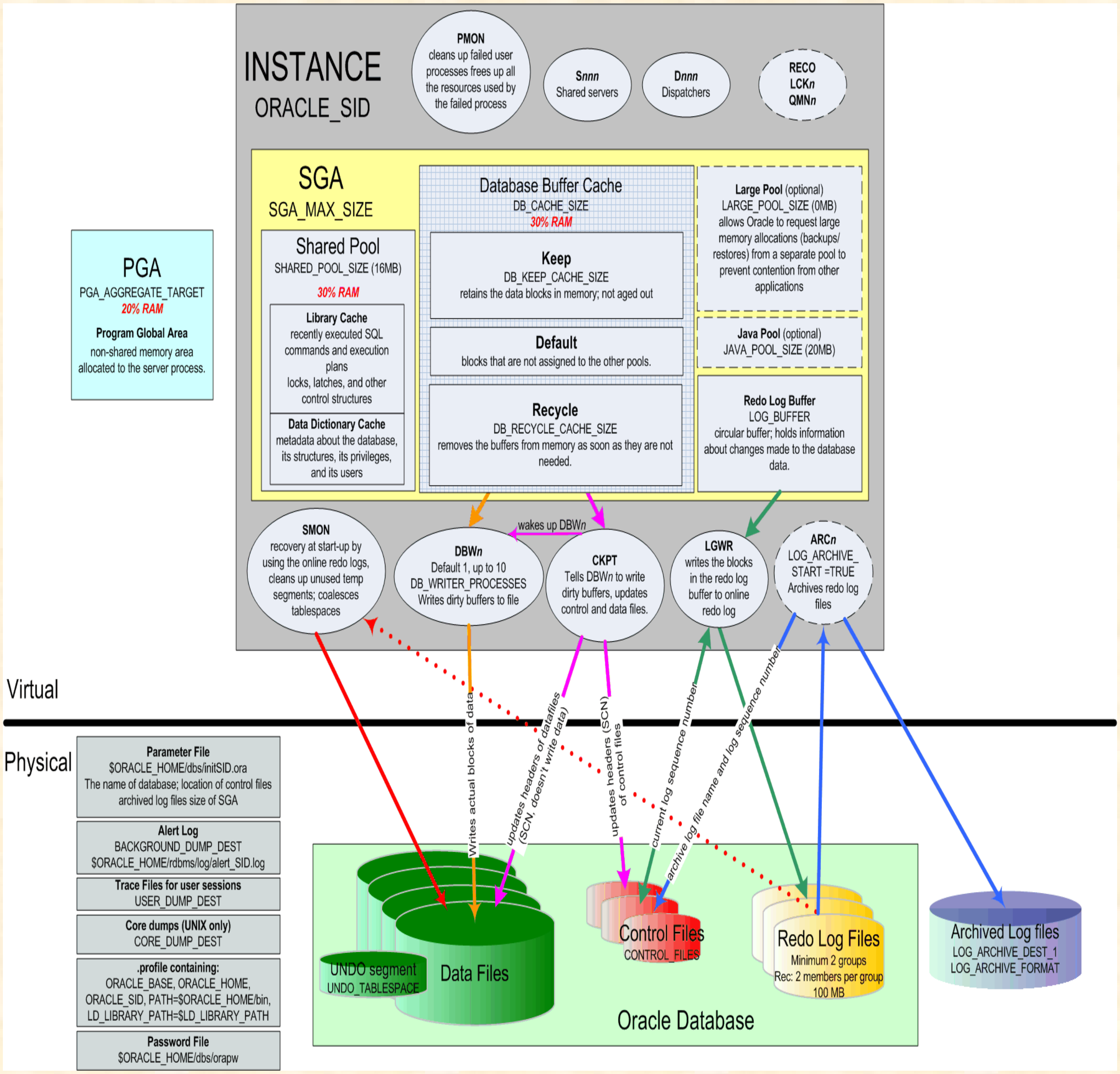
Oracle Architectural Components

عند الحديث عن قواعد البيانات لابد من البدء عن مكونات قاعدة
البيانات وطريقة عملها حتى يتسنى لنا بعد ذلك الحديث عن التفاصيل التى لا
يمكن إستيعابها حتى نفهم تكوينها وطريقة عملها أولاً .

Oracle Database Server



قاعدة البيانات اوركل تتكون من جزئين رئيسيين وهما Oracle Instance و Oracle Database وسنتحدث عن كل منها بالتفصيل فيما بعد .



:Oracle Instance 1.1

وهى تتكون من جزئين رئيسيين كما فى الشكل اعلاه :-

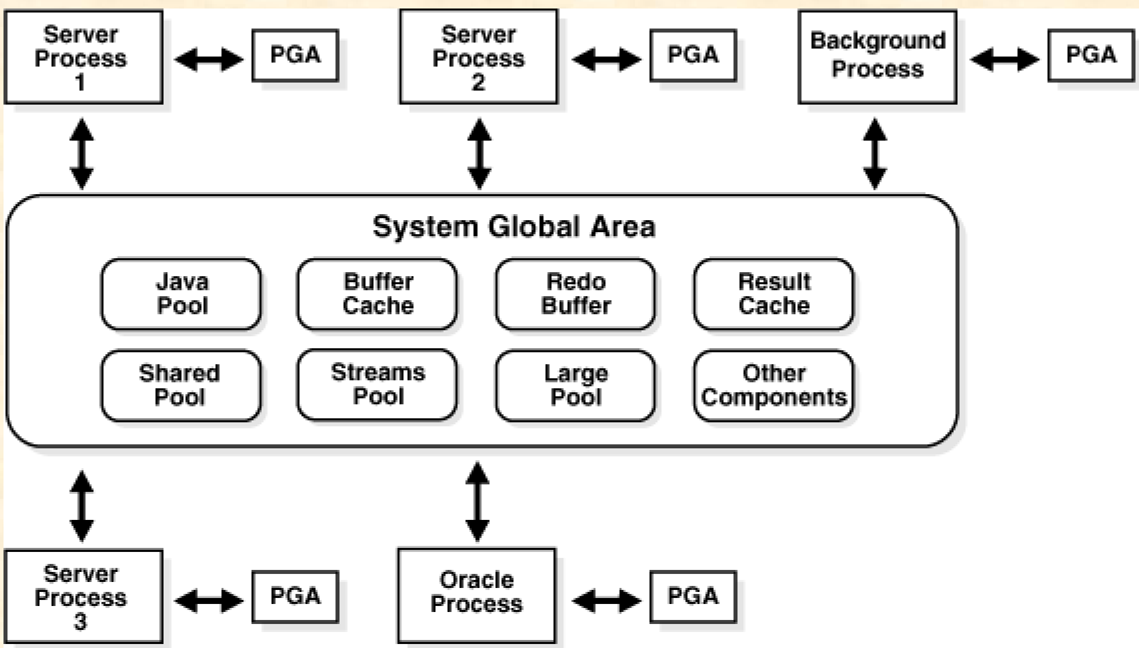
- Memory Structure
- Background Processes

1.1.1 Memory Structure:

وهى تتكون لحظة فتح الInstance وهى عبارة عن جزء من الذاكرة يتم تخصيصه لعمل قاعدة البيانات اوركل وهى تتكون من جزئيين :-

1- System Global Area (SGA)
2- Program Global Area (PGA)

1.1.1.1 System Global Area (SGA):



وتسمى ايضاً Shared Global Area وهى جزء من الذاكرة يخصص للمعلومات التى تكون مشتركة ومتاحة لجميع مستخدمى قواعد البيانات ، وتحتوى على معلومات التحكم التى تستخدم من قبل الOracle Server وهى تتكون فى الVirtual Memory وتتكون لحظة فتح الInstance ، ومقاس هذه الذاكرة يتحدد بواسطة المتغير SGA_MAX_SIZE فى ملف المتغيرات (Parameter File) ، وهى ذاكرة Dynamic أى يمكن تغيير مقاسها دون إغلاق قاعدة البيانات وهى تتكون من قسمين :-

1- Mandatory Memory

2- Optional Memory

Mandatory Memory :

1- Shared Pool: ويتم التحكم فى مقاس هذه الذاكرة بواسطة المتغير SHARED_POOL_SIZE ، وتحتوى على جزئين :

1- Library Cache

2- Data Dictionary Cache

2- Database Buffer Cache : ويتم التحكم فى مقاس هذا الجزء من الذاكرة بواسطة المتغير DB_CACHE_SIZE .

```
ALTER SYSTEM SET DB_CACHE_SIZE = 96M;
```

وهكذا باقى اجزاء الذاكرة يتم التعديل بنفس الطريقة السابقة .

3- Redo Log Buffer ويتم تحديد مقاس هذا الجزء من الذاكرة بواسطة المتغير LOG_BUFFER .

Optional Memory :

1- Large Pool

2- Java Pool

3- Streams Pool

والجدول أدناه يوضح أجزاء الذاكرة SGA :

SGA Component	Size Controlled By	Areas Of Influence	Simple Descriptions
Shared Pool Oracle 6 thru 10g	**SHAREAD_POOL_SIZE**	**Library cache** Shared SQL * Areas Private SQL * Areas PL/SQL * Procedures and Packages	Oracle need to allocate & deallocate memory as SQL or Procedural Code is executed based on the individual needs of users

		Various * Control Structure	sessions and in accordance to the LRU algorithm
		Dictionary Cache Row Cache *	Highly accessed memory structure that provide information on object structure to SQL statements being parsed
Redo Log Buffer Oracle 6 thru 10g	**LOG_BUFFER**	Redo entries *	Holds changes made to data and allows for reconstruction of data in the case of failure
Database Buffer Cache Oracle 6 thru 10g	**DB_2K_CACHE_SIZE** **DB_4K_CACHE_SIZE** **DB_8K_CACHE_SIZE** **DB_16K_CACHE_SIZE** **DB_32K_CACHE_SIZEDB_KEEP_CACHE_SIZE** **DB_RECYCLE_CACHE_SIZE**	Write List * LRU List *	Holds copies of data requested by SQL and reduces requests to disk by having data in memory You may have many different buffer caches that help segregate on usage patterns
Large Pool Form Oracle 8i	**LARGE_POOL_SIZE**	Shared server * Oracle XA * I/O Server Processes Backup & Restore	For large memory allocations
Java Pool From Oracle 8i	**JAVA_POOL_SIZE**	Run stats* Methods* Classes* Session code* Data in JVM	Memory available for the java memory manager to use for all things Java
Streams Pool From Oracle	**STREAMS_POOL_SIZE**	Stream activity*	New to Oracle 10g Memory available for Stream Processing

10g			

يمكن معرفة مقاس الSGA بالنظر فى ملف المتغيرات (Parameters File) ،
او عن طريق كتابة الامر التالى :-

```
SQL> SHOW SGA
```

```
SQL> SHOW SGA

Total System Global Area  171966464 bytes
Fixed Size                   787988 bytes
Variable Size             145488364 bytes
Database Buffers           25165824 bytes
Redo Buffers                 524288 bytes
SQL>
```

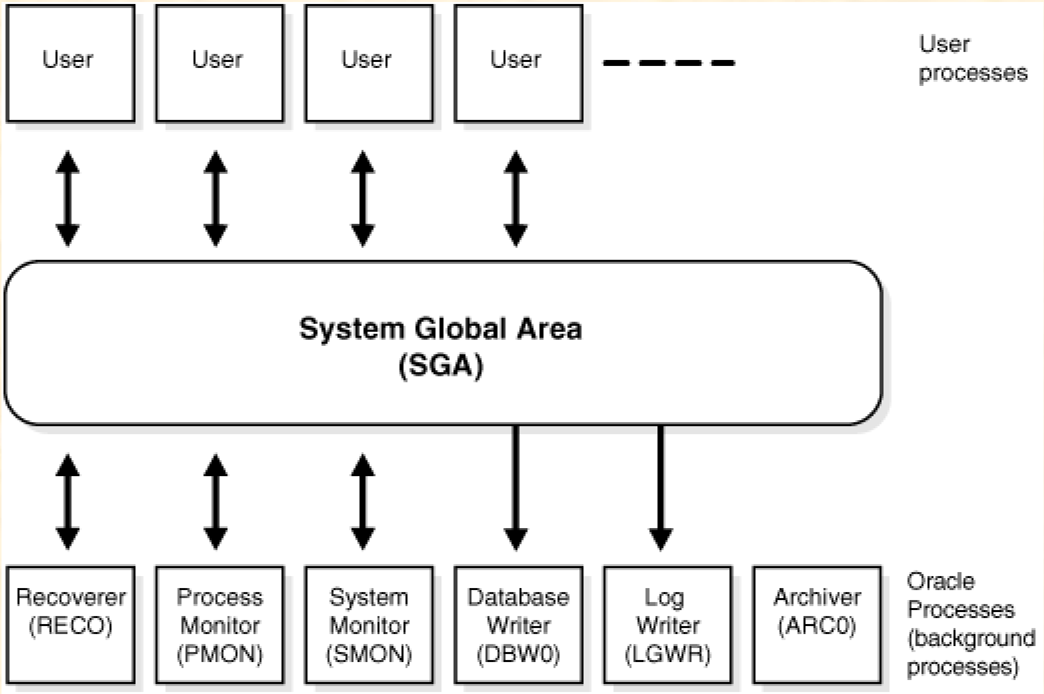
:Related Views
V$SGA *

:Program Global Area (PGA) 1.1.1.2

وتسمى أيضاً Process Global Area وهو جزء من الذاكرة يتكون خارج الInstance وهو يحتوى على معلومات خاصة للServer Process الحالى ويتكون هذا الجزء من الذاكرة لحظة إنشاء الServer Process وتنتهى لحظة إنتهاء الServer Process . وهذا الجزء ليس متاحة لباقى المتصلين أى لكل Server Process فى قاعدة البيانات PGA خاصة به تحتوى على معلومات خاصة به .وهى تحتوى على ثلاثة أجزاء :-

1- Private SQL Area
2- Session Memory
3- SQL Work Area

Background Processes 1.1.2



وقبل الحديث عن ال Background Processes لا بد من الذكر بأن هناك ثلاثة انواع من الProcesses :-

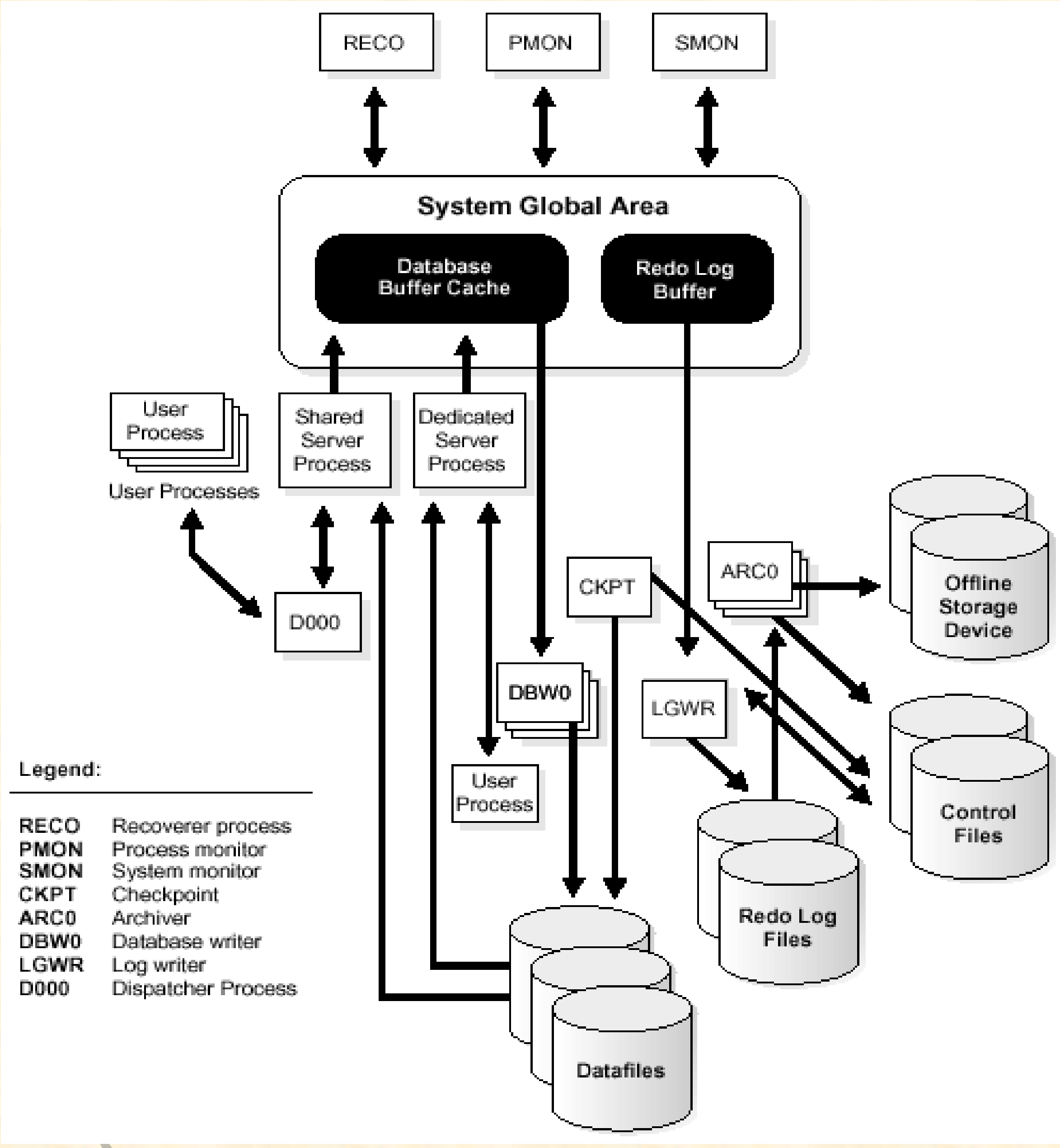
1- User process :- وهو يبدأ العمل عندما يطلب المستخدم الإتصال بقواعد البيانات عن طريق احد ادوات قواعد البيانات .

2- Server Process :- ويتم انشاؤه لحظة الاتصال بالInstances بعد طلب ال User Process الاتصال بقواعد البيانات فيتم التحقق من المستخدم فلحظة الاتصال هى لحظة إنشاء الServer Process وهو يكون بين الUser Process والInstance ، فلكل User Process فى قاعدة البيانات Server Process خاص به هذا إذا كنا نعمل فى بيئة الDedicated Server أما إذا كنا نعمل فى بيئة الShared Server فالأمر يختلف قليلاً ، عموماً سنناقش هذا الامر لاحقاً .

3- Background Processes: - وهو موضوع نقاشنا فى هذه الفقرة وهى عبارة عن معالجات تعمل فى قاعدة البيانات بحيث تقوم بمهام مختلفة تبدأ العمل لحظة فتح الInstance ، وتنقسم الى قسمين:-

1- Mandatory: لا بد من عملها لحظة فتح الInstance .

2- Optional : وبدونها تستطيع الInstance العمل وهذا النوع يعمل فى بعض الاحوال التى يتم فيه تهيئة قاعدة البيانات للعمل على خيارات معينة .



:Mandatory Processes 1.1.2.1

ولا يمكن لقاعدة البيانات العمل دون هذه ال Processes ، وهى:-

: System Monitor (SMON) -1

وأقصى عدد لهذا الProcess فى قاعدة البيانات هو 1 ، ويقوم بعمل الاسترجاع (Recovery) إذا حصل مشكلة فى الInstance ، واذا كنا نعمل على البيئة Real Application Clusters (RAC) وهى عمل اكثر من Instance فى قاعدة البيانات الواحدة فإن الSMON فى الInstance السليمة يستطيع عمل Recovery للInstance الاخرى التى حدث فيها مشكلة .

كذلك يستطيع الSMON عمل تنظيف للSegments المؤقتة التى لم يتم استخدامها من فترة طويلة .

2- Process Monitor (PMON) :

وأقصى عدد لهذا الProcess فى قاعدة البيانات هو 1 ، ويقوم بعمل Recovery للProcess إذا حصلت مشكلة فى الUser Process ، كذلك يقوم تنظيف الDatabase Buffer Cache لإتاحة المصادر فى هذا الجزء من الذاكرة للProcess ، وكذلك يقوم بتسجيل المعلومات حول الInstance وال Dispatcher Processes ، وايضاً يقوم بعمل اختبار للDispatcher Processes والServer Processes ويقوم بعمل إعادة تشغيل فى حالة وجود مشكلة فيهم .

3- Log Writer (LGWR) :

وأقصى عدد لهذا الProcess فى قاعدة البيانات هو 1 ، ويقوم بعمل كتابة للبيانات الموجودة فى الRed Log Buffer ويكتبها فى ال Redo Log Files ، ويقوم بهذه العملية فى الاحوال الاتية :-

1- لحظة عمل Commit .
2- كل ثلاث ثوانى .
3- عندما يمتلئ ثلث الRedo Log Buffer .
4- لحظة عمل DBWn، سنناقش هذا لاحقاً .

كذلك اLGWR يقوم بكتابة التزامن للRedo Log Groups فإذا حدثت مشكلة فى Redo log File فإن الLGWR يقوم بإرسال خطأ لملف Alert Log .

ملاحظة : يجب الانتباه إلى أنه لحظة عمل Commit فإن الLGWR يقوم بكتابة البيانات المثبتة وغيرها الموجودة فى الRedo log Buffer إلى الRedo Log File .

نستفيد من عملية الLGWR فى الاسترجاع إذا حصلت مشكلة فى الInstance .

4- Database Writer (DBWn):

وأقصى عدد لهذا الProcess فى قاعدة البيانات هو 20 ، ويقوم بكتابة البيانات الموجودة فى الDatabase Buffer Cache للDatafiles ، ويمكن تهيئة قاعدة البيانات لتعمل بأكثر من DBWn حسب الحاجة واقصلى عدد 20 .والمتغير الذي يتحكم فى عدد هذا الProcess هو `DB_WRITER_PROCESSES` .

ويعمل هذا الProcess بكتابة البيانات الموجودة فى ال Database Buffer Cache للDatafiles فى الحالات الاتية :-

1- لحظة حدوث الCheckpoint وسنتحدث عن ذلك لاحقاً .
2- كل ثلاث ثوانى .
3- لحظة حدوث الLog Switch وسنتحدث عنه لاحقاً .
4- لحظة إغلاق قاعدة البيانات .
5- لحظة وصول الBlock للقيمة المحددة .
6- لحظة إمتلاء الBuffer .
7- عند عمل الاتى :-

* Tablespace Offline
* Tablespace Read Only
* Table Drop or Truncate
* Tablespace Begin Backup

5- Checkpoint Process (CKPT):

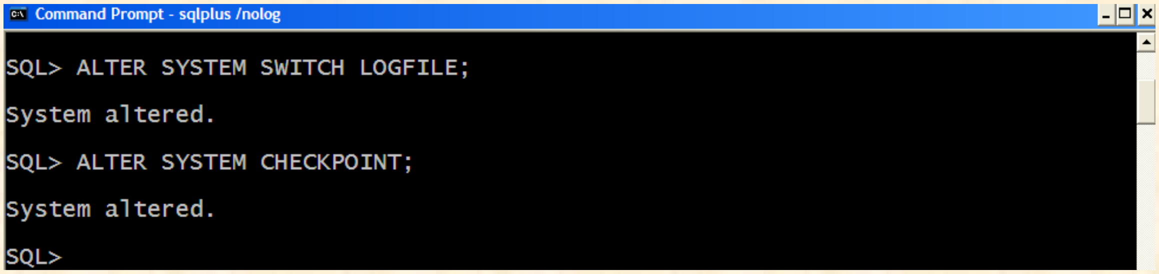
وأقصى عدد لهذا الProcess فى قاعدة البيانات هو 1 ، ويقوم بالتأكد من أن كل التعديلات التى تتم على البيانات فى الBuffer تم كتابتها وتثبيتها فى الDatafiles ومن ثم يقوم بعمل تزامن كامل لكل الDatafiles ويقوم بعمل تعديل للDatafiles headers ؛ والControl files يتم تعديله عند اخر SCN ، بحيث يتم تزامن كامل لقاعدة البيانات ونضمن أنه يمكن استرجاع قاعدة البيانات فى حال حدوث مشكلة .

ويتم عمل الCKPT فى الحالات التالية :-

1- لحظة حدوث Log Switch.
2- عند وصول الزمن المحدد فى المتغير LOG_CHECKPOINT_TIMEOUT
3- عند ما يصل عدد الBLOCKS المحدد فى المتغير LOG_CHECKPOINT_INTERVAL
4- عند وصول عدد الBuffer المحدد فى المتغير FAST_START_IO_TARGET .
5- عند تنفيذ الاوامر التالية :-

```
SQL> ALTER SYSTEM SWITCH LOGFILE;
```

```
SQL> ALTER SYSTEM CHECKPOINT;
```



6- Recover (RECO) :
وأقصى عدد لهذا الProcess فى قاعدة البيانات هو 1 ، وستخدم هذا الProcess لمعالجة مشكلة العمليات الموزعة المعلقة نتيجة مشكلة فى الشبكة او النظام ، فبعد فترة محددة يقوم الProcess بمحاولة الاتصال عن بعد ومحاولة اكمال العملية أو التراجع عنها .

1.1.2.2 Optional Processes:
ويمكن لقاعدة البيانات العمل دون هذه الProcesses ولكثرة هذه ال Processes سنتطرق لاهمها :-

1- Archiver (ARCn) :
وأقصى عدد لهذا الProcess فى قاعدة البيانات هو 10 ، ويقوم بكتابة الOnline Red log Files فى ملف الارشيف (Archive Log Destination) بعد حدوث الLog Switch ، هذا الProcess يعمل إذا كانت قاعدة البيانات تعمل فى النمط Archive Log Mode , يتم التحكم فى عدد هذا الProcess عن طريق المتغير LOG_ARCHIVE_MAX_PROCESSES .

2- Recovery Writer(RVWR):
هذا الProcess تم استحداثه فى الاصدارة Oracle 10g نستفيد من هذا الProcess عند عملية الFlashback Database ، سنناقش هذا الموضوع لاحقاً .

3- Lock Monitor (LMON).
4- Lock Manager DAEMON .
5- Lock Process (LCKn) .
6- Block Server Process (BSPn) .
7- Queue Monitor (QMNn) .
8- Event Monitor (EMNn) .

9- Shared Server Processes (Snnn) .
10- Memory Manager (MMAN) .
11- Parallel Execution slaves (Pnnn).
12- Trace Writer (TRWR) .
13- DMON .
14- Dispatcher (Dnnn) .
15- MMON .
16- Wakeup Monitor Process (WMON) .
17- Memory Monitor Light (MMON) .
18- RBAL .
19- ARBx .
20- ASMB .
21- Change Tracking Writer (CTWR) .
22- Job Queue Monitoring (CJQn) .

1.2 Oracle Database:

وهو الجزء الثانى من مكونات قاعدة البيانات اوركل كما ذكرنا سابقاً :-

Oracle Database server = Oracle Instance + oracle Database

ويحتوى هذا الجزء من مجموعة من الملفات :-

1- Control files :-

وهو الملف المسؤل عن التزامن فى قاعدة البيانات بجانب انه يحتوى عن المعلومات الاساسية عن قاعدة البيانات كإسم قاعدة البيانات وبدونه لا تعمل قاعدة البيانات إذ أنه يحتوى على مسارات ملفات قاعدة البيانات وإذا تمت إضافة ملف فى قاعدة البيانات يتم تحديث الـControl file اوتومتيكياً .

2- Redo log files :-

ويستخدم هذا الملف حتى نستطيع عمل إسترجاع لقاعدة البيانات (Recover) فى حالة حدوث مشكلة فى قاعدة البيانات ، إذ أنه يحفظ التغيرات التى تحدث فى قاعدة البيانات .
يتم تحديد هذا الملف عند فتح قاعدة البيانات عن طريق الـParameter file ، ولحمايته من الفقدان يجب استخدام اكثر من نسخة من هذا الملف عن طريق تحديد هذه النسخ فى ملف الـParameter file .

3- Database files :-

وهو المخزن الحقيقى للبيانات فى قاعدة البيانات أى أنه يحتوى على الجداول والمناظير والمراجع (Table & Views & Indexes) والكائنات الاخرى .

4- **Archived redo log files** :-

وهو عبارة عن نسخة من ال Redo log files ونحتاجه أيضاً عند عمل إسترجاع لقاعدة البيانات (Recover) .

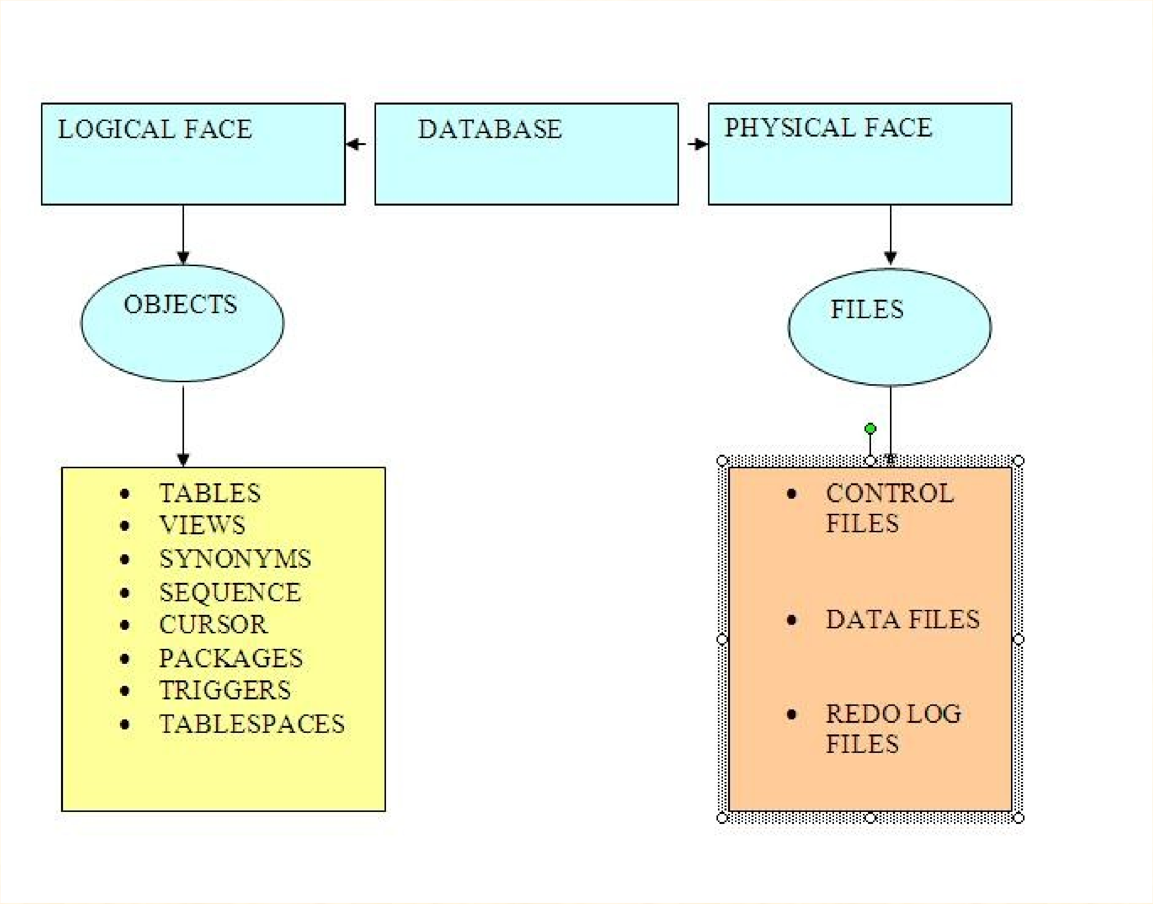
5- **الملفات الأخرى**:-

* **Parameter file** ويستخدم لعمل تهيئة للOracle Instance لحظة تشغيلها

* **Password file** ويسمح هذا الملف للمستخدمين الأتصال بقاعدة البيانات عن بعد كمدراء لقاعدة البيانات ، وليس لتخزين كلمات السر للمستخدمين فى قاعدة البيانات كما يتصور البعض .

Tablespace and Data File:-

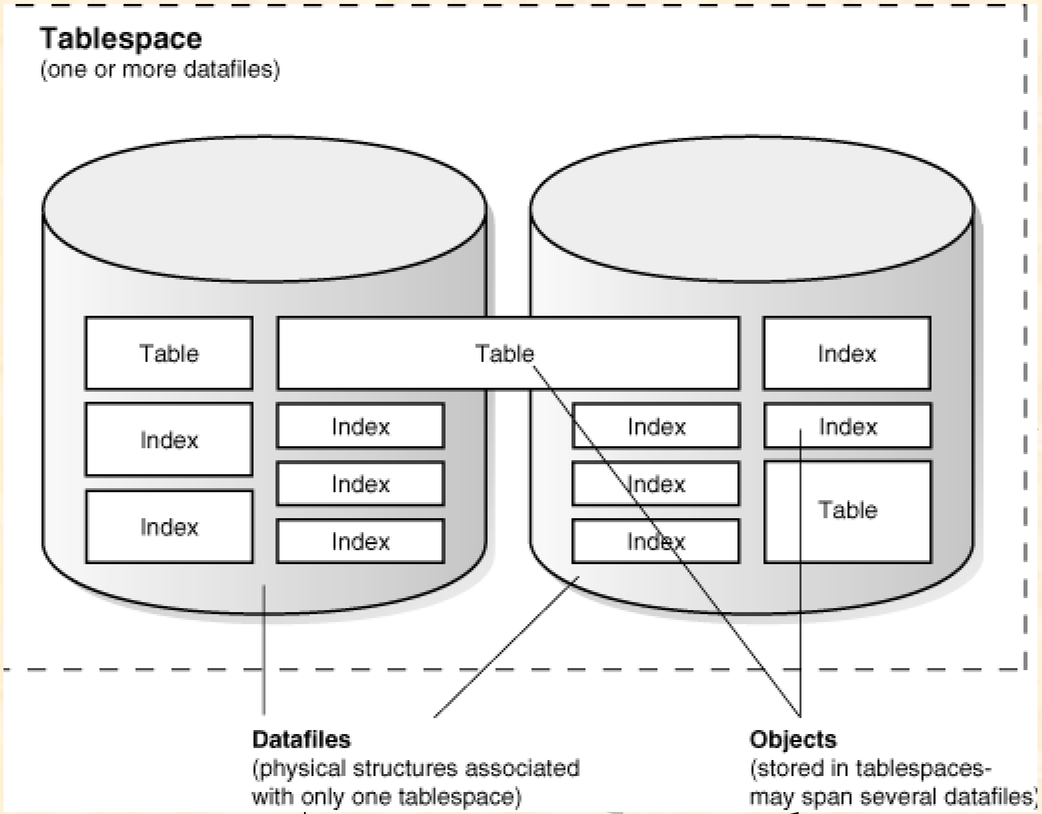
وهنا لابد من الحديث إلى انه يمكن تقسيم قاعدة البيانات الى (Physical & Logical) :



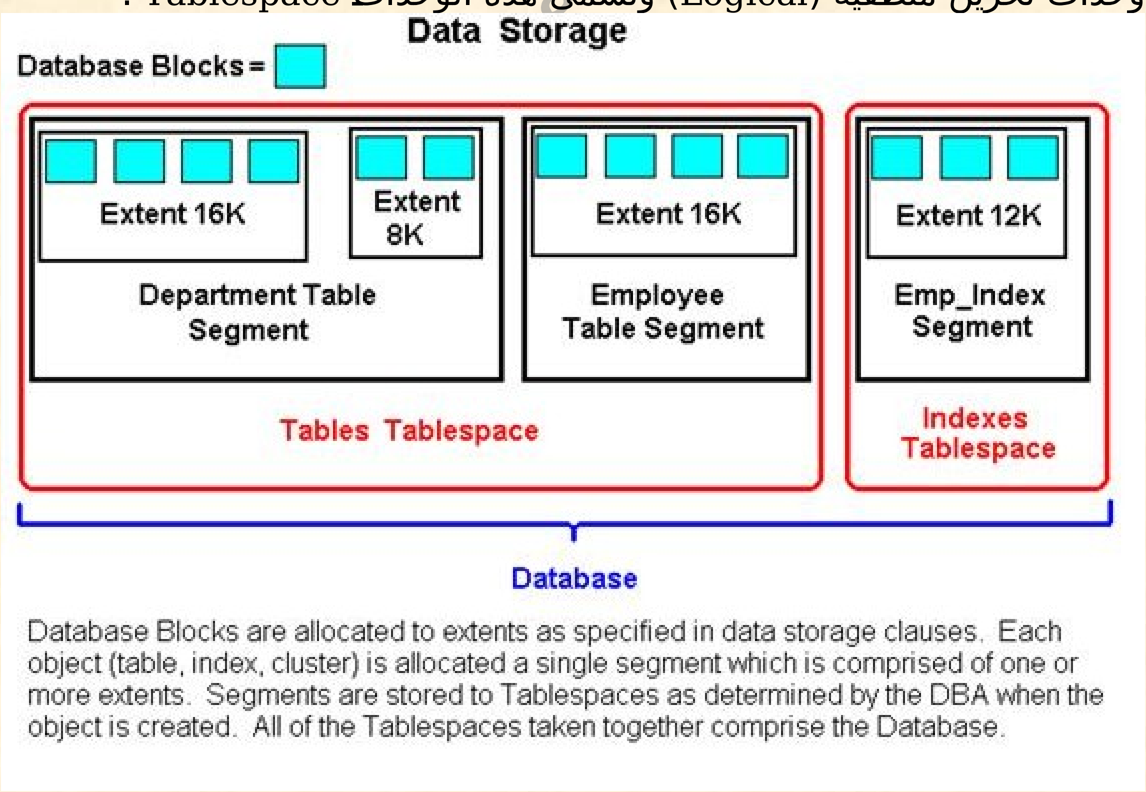
Physical: حيث يمكن التعامل مع هذا النوع سواء كانت قاعدة البيانات مفتوحة أو مغلقة بحيث يمكن التعامل مع هذا النوع عن طريق نظام التشغيل أو عن طريق قاعدة البيانات.

Logical: ولا نستطيع التعامل مع هذا النوع إلى أذا كانت قاعدة البيانات مفتوحة.

-:Tablespace and Data File

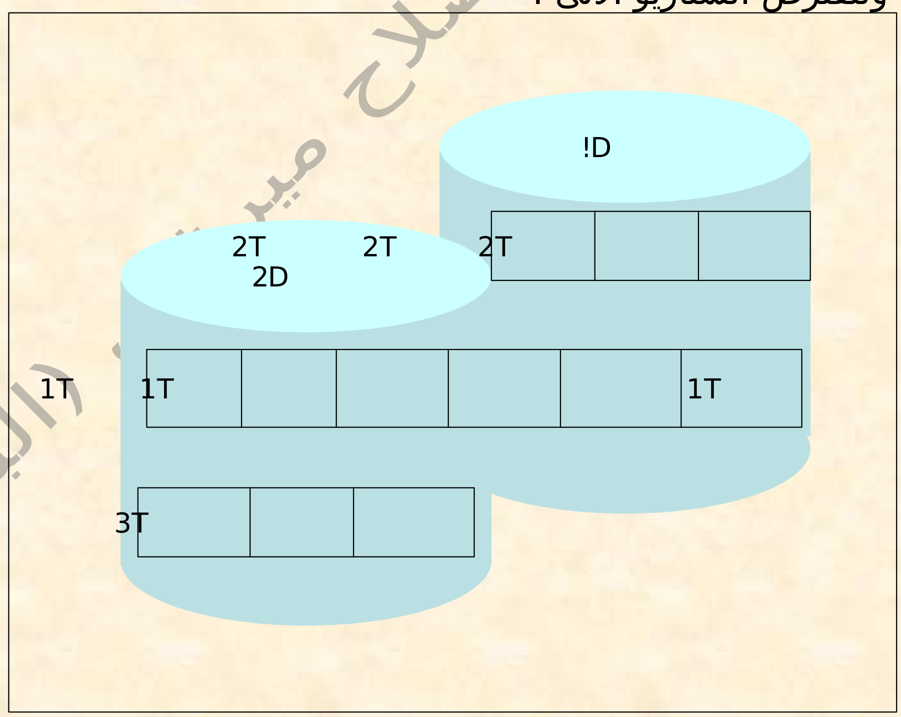


قاعدة البيانات فيزيائياً تتقسم إلى ملفات تخزينية يمكن أن تشاهد عن طريق نظام التشغيل وهى ملفات الData files ، وكذلك قاعدة البيانات تنقسم إلى وحدات تخزين منطقية (Logical) وتسمى هذه الوحدات Tablespace .



الTablespace وهو وحدة منطقية يحتوى على Data file واحد أو أكثر وكل Data file ينتمى فى الحقيقة إلى Tablespace واحد .
يمكن إنشاء Bigfile Tablespaces بحيث يحتوى على ملف Data file واحد ولكن كبير جداً بحيث يكون اكبر من 4GB .
الTablespace يتكون من مجموعة من الوحدات المنطقية الاصغر وهى تسمى Segments وهى وحدات منطقية يتكون منها الTablespace بحيث تنتمى ال Segment لTablespace واحد فقط وهى تتكون من مجموعة من الوحدات المنطقية الأصغر التى تسمى Extents وهى ايضاً وحدات منطقية تنتمى كل Extent لSegment واحد وتتكون أيضا الExtents من مجموعة أخرى أصغر وهى أصغر وحدة منطقية تسمى الBlocks .
وعند إنشاء قاعدة البيانات يتم تحديد حجم الBlock لقاعدة البيانات ، ويمكن أن يكون حجم الBlock 3BK او 5BK أو غيره وأكبر حجم للBlock يتوقف على نظام التشغيل ، وقبل الاصدار Oracle 9i Release كانت قاعدة البيانات تعمل على حجم واحد للBlock وابتداءً من الإصدار Oracle 9i Release أمكن لقاعدة البيانات العمل بأحجام مختلفة من اللBlock

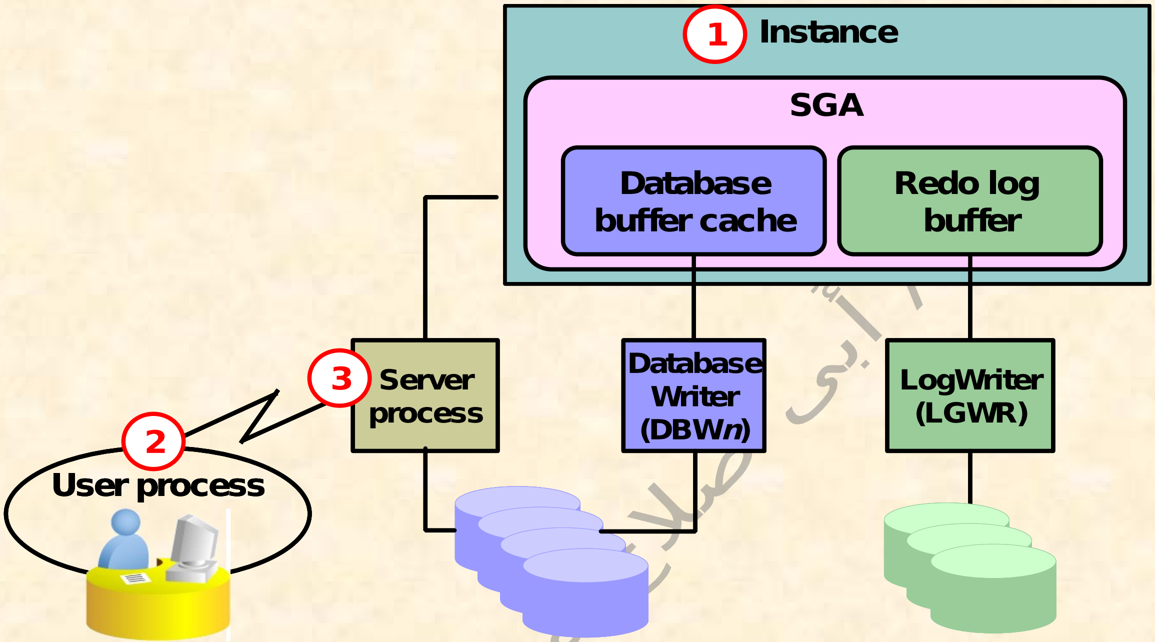
ولنفترض السناريو الاتى :-



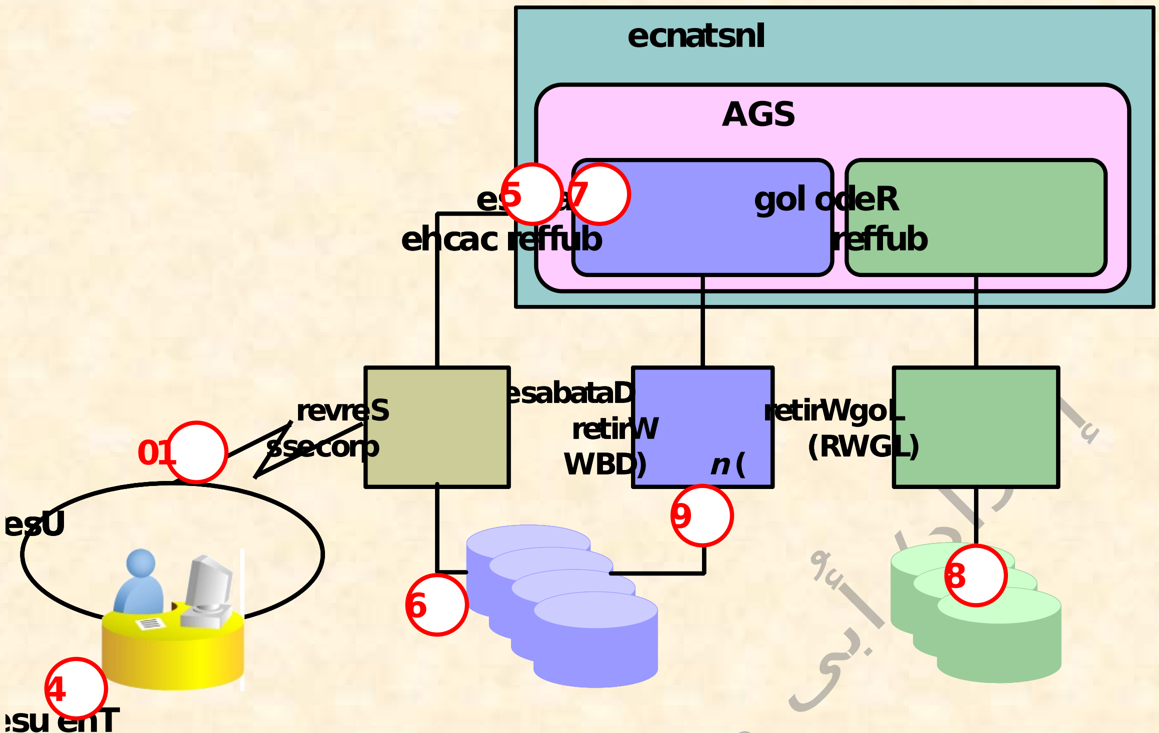
فانفترض أن هذا الشكل يمثل Tablepace يسمى Users . هذا الTablespace يحتوى على ملفين من ال(D1&D2 (Data fileهذا الTablespace يحتوى على مجموعة من ال(T1&T2&T3) Segments الSegment الاول T1 مقسم إلى مجموعة من الExtents   وكل Extent بالطبع مكون من مجموعة من ال Blocks ، نلاحظ هنا أن الT1 Segment يمتد على كل من الملفين (D1&D2)

أى أن جزء من الSegment فى الملف D1 والجزء الاخر فى الملف D2 لكن فى الوقت نفسه هو ينتمى لTablespace واحد .

قبل نهاية هذا الباب فلنتابع معاً هذا السناريو وهو يوضح خطوات عمل قاعدة البيانات :



1- الInstance تعمل على المخدم (Server).
2- فى هذه المرحلة المستخدم يحاول الاتصال بقاعدة البيانات عن طريق ال Application أو أحد أدوات قواعد البيانات.
3- فى هذه اللحظة تم التحقق من طلب المستخدم وتم إنشاء الاتصال وتكوين Server Process.



4- هنا المستخدم طلب تعديل صف .
5- الServer Process يستقبل هذا الطلب ويقوم بعمل اختبار للShared Pool هل هذا الطلب موجود فى الShared SQL Area إذا كان موجود يقوم بالتأكد من أن للمستخدم صلاحية الوصول لهذا البيانات ، اما إذا كان هذا الطلب غير موجود يقوم بإنشاء Shared SQL Area جديد .
6- فى هذه المرحلة يقوم الServer Process بجلب البيانات المطلوبة من الData file من الجدول أو من الData Block المخزنة فى الSGA .
7- بعد جلب البيانات هنا يقوم الServer بتعديل الجدول فى الSGA .
8- لحظة عمل Commit يقوم الLGWR بكتابة العملية فى الRedo Log File .
9- يقوم الDBWn بكتابة التعديلات فى الDisk اى فى الData File .
10- اخيراً يرسل الServer Process يرسل رسالة بنجاح أو فشل العملية .

الفصل الثانى
إنشاء قاعدة البيانات أوركل
Creating an Oracle Database

النقاط الرئيسية التى سنناقشها فى هذا الباب :-

- **فهم الشروط الضرورية لإنشاء قاعدة البيانات .**
- **إنشاء قاعدة بيانات بالطريقة اليدوية .**
- **إنشاء قاعدة البيانات بالأداة DBCA .**

من أهم مهام مدير قاعدة البيانات هو التخطيط لقاعدة البيانات ، إذ لا يتصور مديراً ناجحاً لقاعدة البيانات لا يخطط بصورةٍ ما لقاعدة بياناته ، فما نوع البيانات التى سوف نخزنها فى كل Tablespace وكم Data file فى كل Tablespace وكيف سيتم تخزين ملفات قاعدة البيانات فيزيائياً على الديسك ، وكيف سيتم عمل نسخ احتياطى لقاعدة البيانات ، وكيف نحافظ على قاعدة البيانات ، وكيف نستطيع رفع الأداءة فى قاعدة البيانات وغيرها من الاسئلة التى يجب ان تجد لها جواباً .

بعد ذلك يجب أن نحدد ما نحتاجه من الذاكرة والديسك المناسب حسب متطلبات قاعدة البيانات وعموما يجب وضع هذه الاعتبارات عند إنشاء قاعدة البيانات .

فى الحقيقة يمكن إنشاء قاعدة البيانات أثناء إنزال الاوركل عن طريق ال OUI (Oracle Universal Installer) وذلك بواسطة الاداة DBCA ولكن قد نحتاج لإنشاء قاعدة بيانات أخرى ،.

عند إنشاء قاعدة بيانات اوركل يجب مراعاة النقاط الاتية:-

1- كم عدد الApplication التى تعمل على قاعدة البيانات .

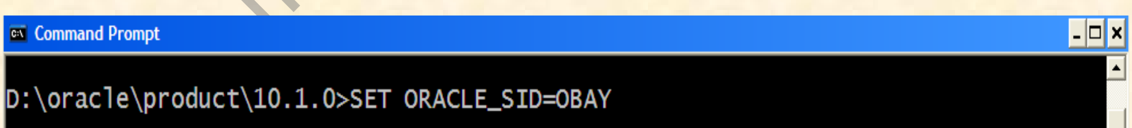
2- عدد المستخدمين إذ نحتاج وضع المتغير process فى ملف المتغيرات (parameter file) .
3- مساحة الذاكرة SGA فهذه الذاكرة مطلوبة لعمل الInstance وهى لا تقبل المشاركة بين أكثر من Instance أى لكل Instance ذاكرة SGA خاصة بها .
4- كذلك الBlock_Size الذى يعتمد عليه الRow_Size .
5- هل يمكن إغلاق قاعدة البيانات لعمل النسخ الإحتياطى .

1- إنشاء قاعدة البيانات بالطريقة اليدوية (Manually) :

قبل البدء يجب الإشارة إلى أننا سنستخدم نظام التشغيل windows .

1- تحديد اسم ال(ORACLE_SID) Instance :- فقد يكون هناك أكثر من Oracle Instance فى الجهاز الواحد وذلك باستخدام المتغير ORACLE_SID . اسم الInstance هنا OBAY .

```
D:\ORACLE\PRODUCT\10.2.0>SET ORACLE_SID=OBAY
```



2- إنشاء Oracle Service وذلك لأننا نعمل على نظام التشغيل WINDOWS إذ نحتاج للService لكل Instance تعمل فى نظام التشغيل WINDOWS ولا يلزمنا ذلك فى نظام التشغيل LINUX .
وهى بإختصار عبارة عن SERVICE يتم إنشاؤها فى الWINDOWS .

```
D:\ORACLE\PRODUCT\10.2.0\DB_1>ORADIM –NEW –SID
%ORACLE_SID% -INTPWD PASSWORD –STARTMODE M
```

يمكن التحقق من الإنشاء عن طريق نظام التشغيل بالذهاب الى الٍServices .

```
D:\oracle\product\10.1.0\db_1>ORADIM -NEW -SID %ORACLE_SID% -INTPWD PASSWORD -ST
ARTMODE M
Instance created.

D:\oracle\product\10.1.0\db_1>_
```

3- إنشاء ملف المتغيرات (Parameter File) -: وذلك بنسخ الملف من المسار التالى :

```
D:\ORACLE\PRODUCT\10.2.0\db_1\database\copy
init.ora  INITobay.ora
```

اما إذا كنا نعمل على نظام التشغيل لينكس (UNUX) فإن المسار سيكون :
$ORACLE_HOME/DB
نتابع الخطوات السابقة :

```
D:\oracle\product\10.1.0>SET ORACLE_SID=OBAY

D:\oracle\product\10.1.0>ORADIM -NEW -SID %ORACLE_SID% -INTPWD PASSWORD -STARTMODE M
Instance created.

D:\oracle\product\10.1.0>CD db_1

D:\oracle\product\10.1.0\db_1>CD database

D:\oracle\product\10.1.0\db_1\database>DIR
 Volume in drive D has no label.
 Volume Serial Number is A075-24C3

 Directory of D:\oracle\product\10.1.0\db_1\database

03/11/2008  08:58 PM    <DIR>          .
03/11/2008  08:58 PM    <DIR>          ..
03/11/2008  02:53 PM    <DIR>          archive
03/11/2008  04:45 PM               106 core_test_4024_3604_2008_3_11_15_22_45.log
03/11/2008  04:45 PM             2,048 hc_obay.dat
03/11/2008  03:11 PM             2,048 hc_orcl.dat
03/11/2008  02:40 PM            13,212 initdw.ora
03/11/2008  08:54 PM               955 INITorcl.ORA
03/11/2008  02:40 PM            31,744 oradba.exe
03/11/2008  08:58 PM             2,560 PWDOBAY.ORA
03/11/2008  03:00 PM             1,536 PWDorcl.ora
03/11/2008  02:57 PM             2,560 SPFILEORCL.ORA
               9 File(s)         56,769 bytes
               3 Dir(s)  94,657,060,864 bytes free

D:\oracle\product\10.1.0\db_1\database>COPY INITorcl.ORA INITobay.ORA
        1 file(s) copied.

D:\oracle\product\10.1.0\db_1\database>
```

بالطبع نحتاج لتعديل ملف المتغيرات حسب المعطيات الجديدة ، على سبيل المثال
-:

```
CONTROL_FILES = (‘D:\oracle\product\10.1.0\
oradata\OBAY\CONTROL01.CTL,
D:\oracle\product\10.1.0\oradata\OBAY\
CONTROL02.CTL,
D:\oracle\product\10.1.0\oradata\OBAY\
CONTROL03.CTL)
UNDO_MANAGEMENT = AUTO
undo_tablespace = ‘UNDO'
DB_NAME         = OBAY
DB_BLOCK_SIZE   = 8192
background_dump_dest = 'D:\oracle\product\10.1.0\
admin\obay\bdump’
user_dump_dest       = 'D:\oracle\product\10.1.0\
admin\obay\udump'
core_dump_dest       = 'D:\oracle\product\10.1.0\
admin\obay\cdump'
```

4- تشغيل الInstance فى الوضع NOMOUNT

```
Command Prompt - SQLPLUS /NOLOG

D:\oracle\product\10.1.0\db_1>SQLPLUS /NOLOG

SQL*Plus: Release 10.1.0.2.0 - Production on Sat May 10 03:16:04 2008

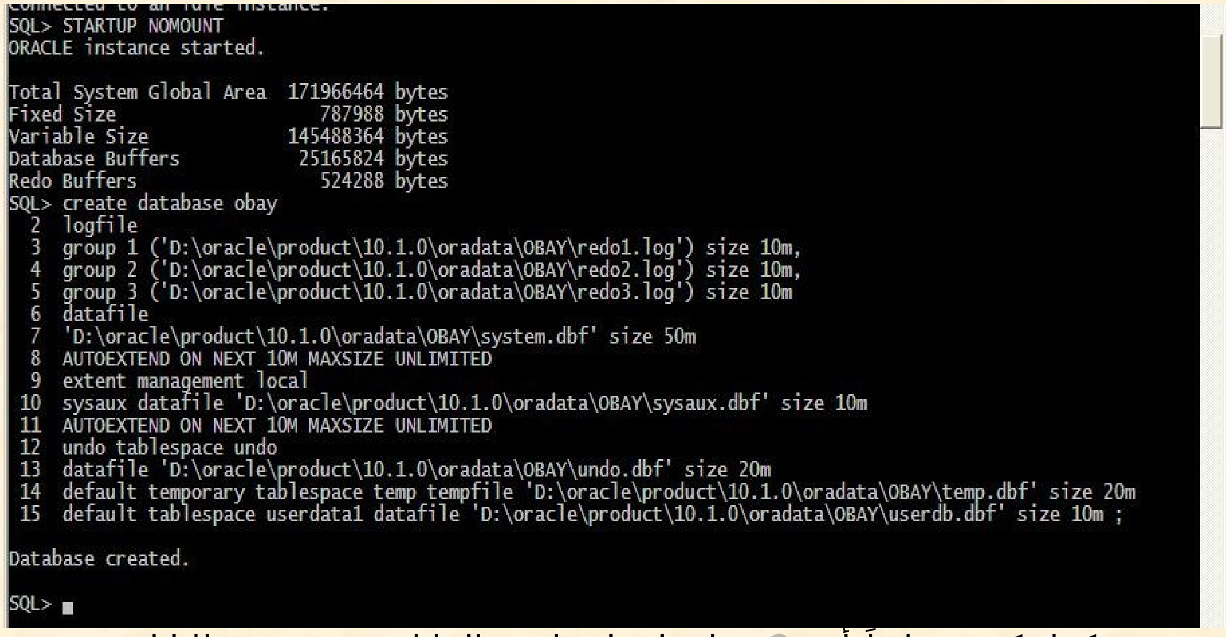
Copyright (c) 1982, 2004, Oracle.  All rights reserved.

SQL> CONN /AS SYSDBA
Connected to an idle instance.
SQL> STARTUP NOMOUNT
ORACLE instance started.

Total System Global Area  171966464 bytes
Fixed Size                   787988 bytes
Variable Size             145488364 bytes
Database Buffers           25165824 bytes
Redo Buffers                 524288 bytes
SQL> _
```

5- الان نقوم بإنشاء قاعدة البيانات

```
create database obay
Logfile
group 1 ('D:\oracle\product\10.1.0\oradata\OBAY\
redo1.log') size 10m,
group 2 ('D:\oracle\product\10.1.0\oradata\OBAY\
redo2.log') size 10m,
group 3 ('D:\oracle\product\10.1.0\oradata\OBAY\
redo3.log') size 10m
datafile
'd:\obay\system.dbf' size 50m
AUTOEXTEND ON NEXT 10M MAXSIZE UNLIMITED
extent management local
sysaux datafile 'D:\oracle\product\10.1.0\oradata\
OBAY\sysaux.dbf' size 10m
AUTOEXTEND ON NEXT 10M MAXSIZE UNLIMITED
undo tablespace undo
datafile 'D:\oracle\product\10.1.0\oradata\OBAY\
undo.dbf' size 20m
default temporary tablespace temp tempfile 'D:\
oracle\product\10.1.0\oradata\OBAY\temp.dbf' size
20m
default tablespace userdata1 datafile 'D:\oracle\
product\10.1.0\oradata\OBAY\userdb.dbf' size 10m ;
```



```
SQL> STARTUP NOMOUNT
ORACLE instance started.

Total System Global Area  171966464 bytes
Fixed Size                   787988 bytes
Variable Size             145488364 bytes
Database Buffers           25165824 bytes
Redo Buffers                 524288 bytes
SQL> create database obay
  2  logfile
  3  group 1 ('D:\oracle\product\10.1.0\oradata\OBAY\redo1.log') size 10m,
  4  group 2 ('D:\oracle\product\10.1.0\oradata\OBAY\redo2.log') size 10m,
  5  group 3 ('D:\oracle\product\10.1.0\oradata\OBAY\redo3.log') size 10m
  6  datafile
  7  'D:\oracle\product\10.1.0\oradata\OBAY\system.dbf' size 50m
  8  AUTOEXTEND ON NEXT 10M MAXSIZE UNLIMITED
  9  extent management local
 10  sysaux datafile 'D:\oracle\product\10.1.0\oradata\OBAY\sysaux.dbf' size 10m
 11  AUTOEXTEND ON NEXT 10M MAXSIZE UNLIMITED
 12  undo tablespace undo
 13  datafile 'D:\oracle\product\10.1.0\oradata\OBAY\undo.dbf' size 20m
 14  default temporary tablespace temp tempfile 'D:\oracle\product\10.1.0\oradata\OBAY\temp.dbf' size 20m
 15  default tablespace userdata1 datafile 'D:\oracle\product\10.1.0\oradata\OBAY\userdb.dbf' size 10m ;

Database created.

SQL>
```

كما ذكرت سابقاً أنت تستطيع إنشاء قاعدة البيانات حسب متطلباتك الخاصة وليست فقط كما هو مذكور فى النموزج أعلاه , لكن لابد من إنشاء (SYSTEM & SYSAUX TABLESPACE) .

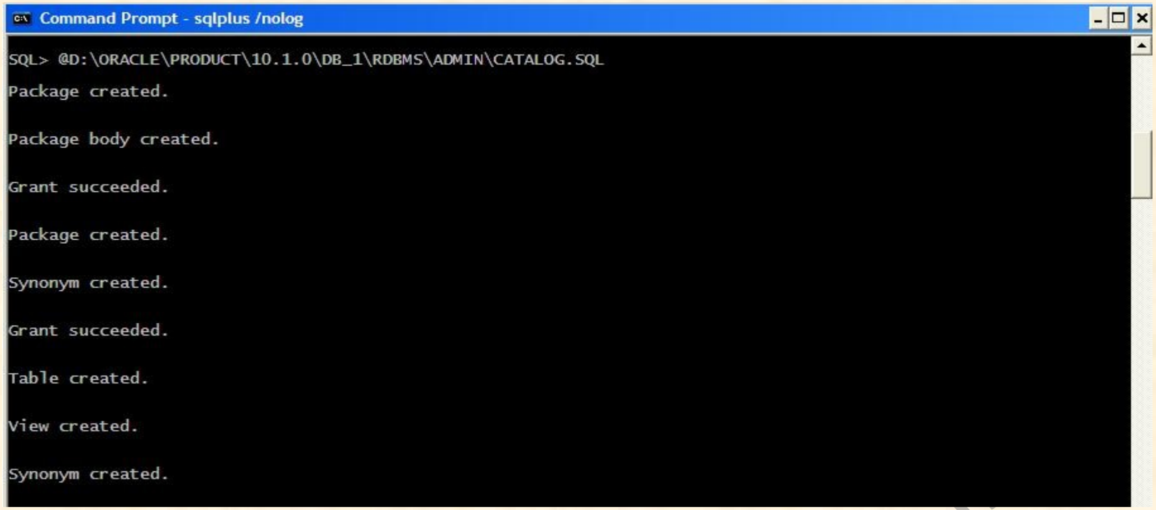
إذا حدث خطأ اثناء إنشاء قاعدة البيانات فإن الخطأ سيكتب فى الملف Alert Log الموجود فى المسار المحدد فى ملف المتغيرات بالمتغير ال BACKGROUND_DUMP_DEST .

أما إذا ظهرت الرسالة ORA-01031 اثناء إنشاء قاعدة البيانات فهذا يعنى ان مستخدم نظام التشغيل ليس عضواً فى المجموعة ORA_DBA فيجب إضافته فى المجموعة .

بعد إنشاء قاعدة البيانات يمكن تشغيلها فى الوضع MOUNT او فتحها للإستخدام .

6- إنشاء الData Dictionary :- وذلك من خلال تشغيل الملف فى المسار التالى :-

D:\ORACLE\PRODUCT\10.1.0\DB_1\RDBMS\ADMIN\



ولكن يجب تشغيل الملف على المستخدم SYS .

كذلك نشغل الملف على المسار التالى :-

```
D:\ORACLE\PRODUCT\10.1.0\DB_1\RDBMS\ADMIN\
```

وذلك لإنشاء كل الStructures المطلوبة لعمل PL/SQL .

لا يختلف الامر كثيراً عند إستخدام نظام التشغيل (UNIX)

2- إنشاء قاعدة البيانات عن طريق ال(DBCA) :

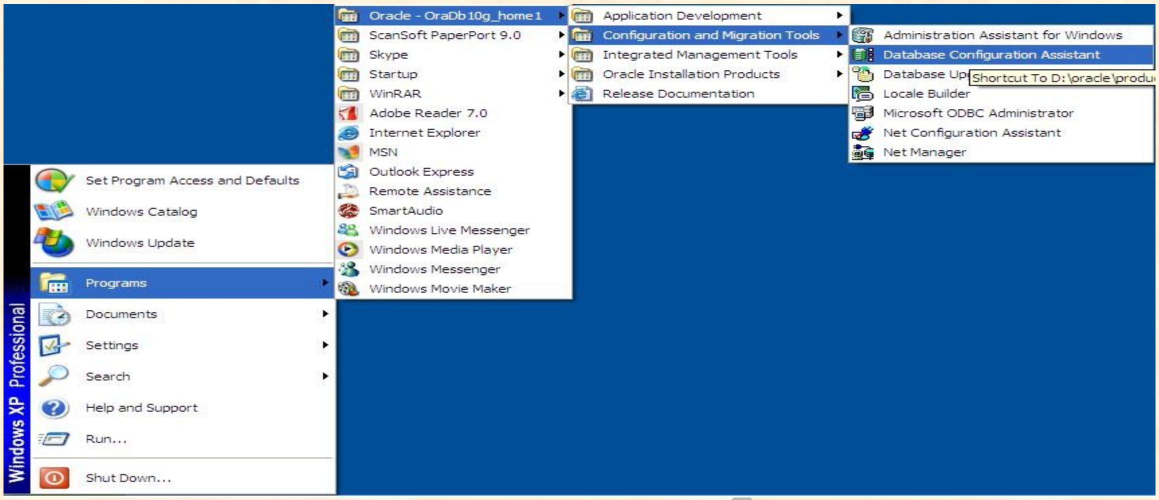
وهى أداة اصدرتها اوركل لعدة اغراض :-

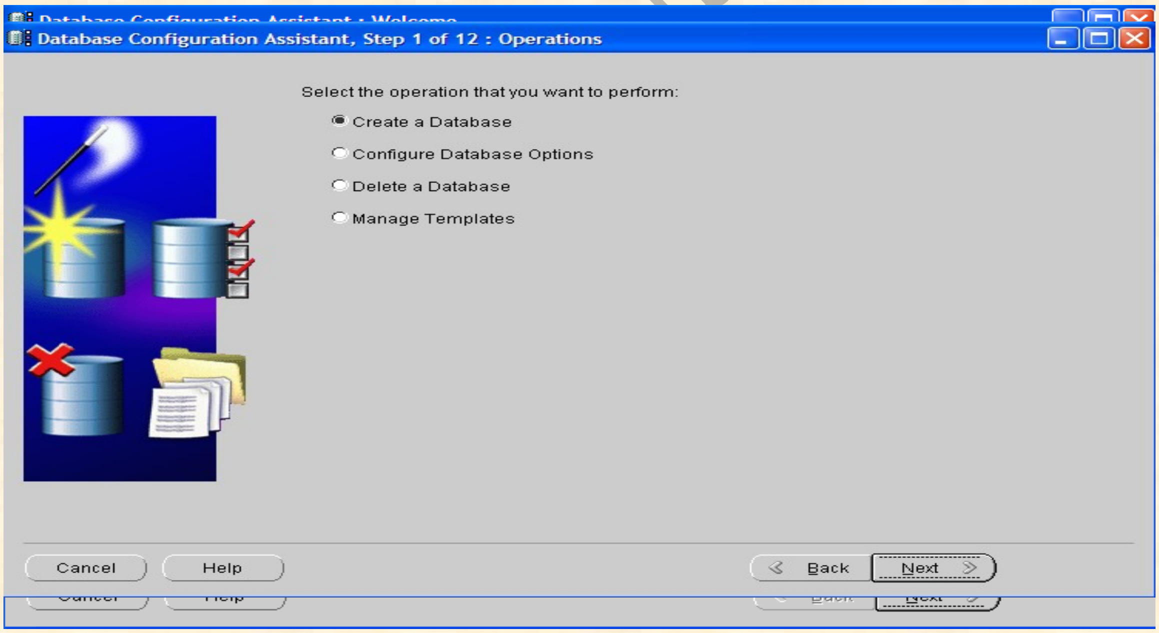
1- إنشاء قاعدة البيانات .

2- إعادة تهيئة قاعدة البيانات .

3- حذف قاعدة البيانات .
4- إنشاء قوالب لقاعدة البيانات (Templates) .

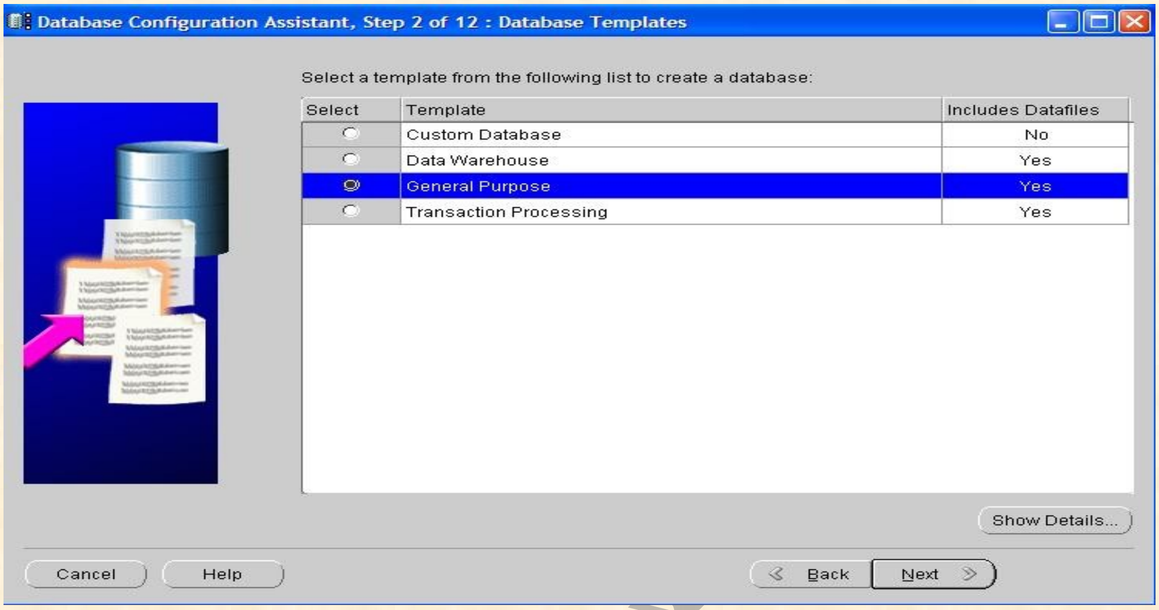
نتابع خطوات إنشاء قاعدة بيانات اوركل عن طريق الأداة
DATABASE CONFIGURATION ASSISTANT (DBCA)



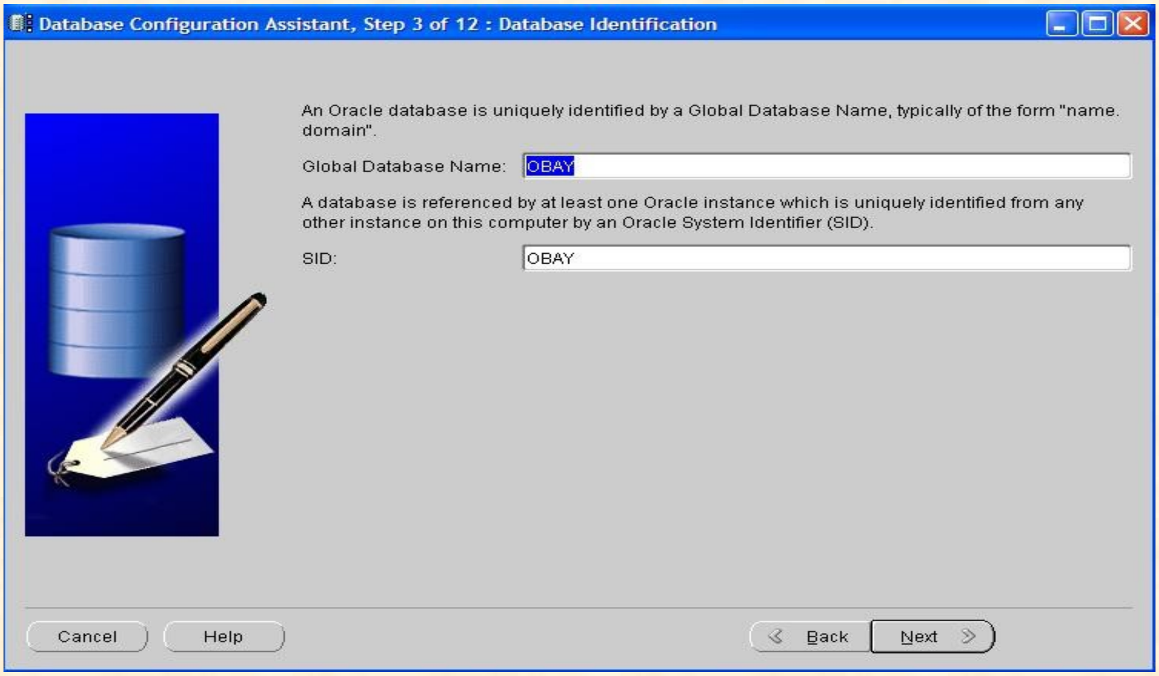


هنا الخيارات المتاحة للمستخدم ، سنختار بالطبع الخيار الاول إنشاء قاعدة بيانات .
إذا اخترت الخيار الثانى سيعرض لى قواعد البيانات الموجودة عندى ومن ثم اختار قاعدة البيانات المراد إعادة تهيئتها ، وبعد ذلك سيعرض لى بعض الخيارات لإعادة تهيئتها .

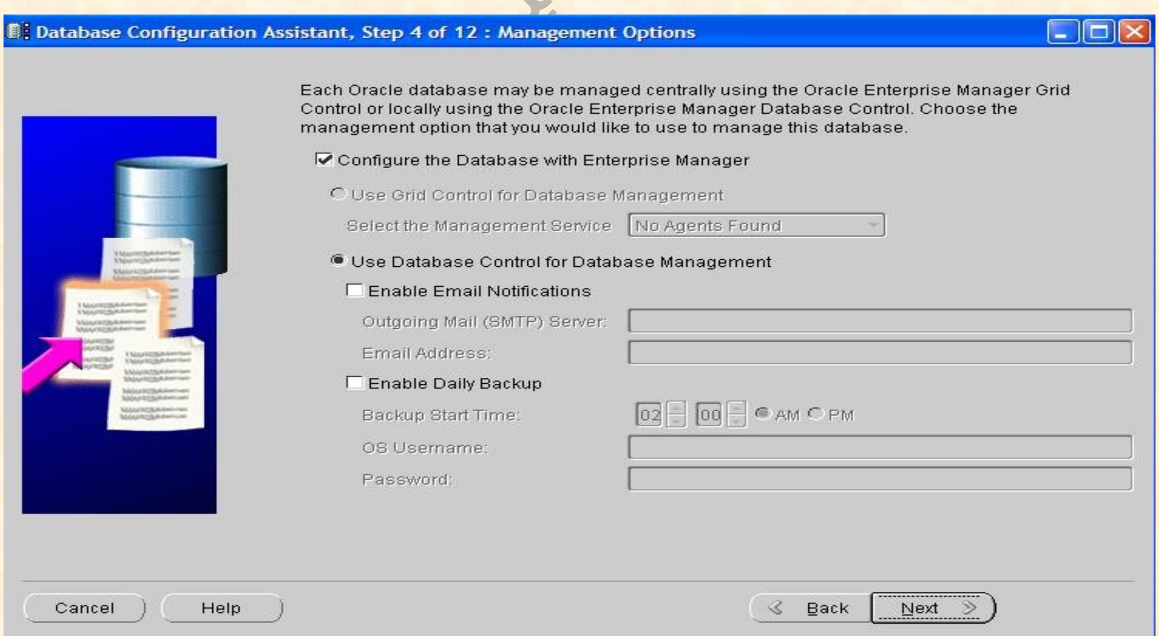
الخيار الثالث بالطبع استطيع من خلاله خذف قاعدة البيانات ، فقط ما على إلا أن اختار قاعدة البيانات المراد حزفها .
الخيار الاخير هو إما لأنشاء قالب جديد وهو القالب الذى ستكون عليه قاعدة البيانات (Template) أو حذف قالب موجود .



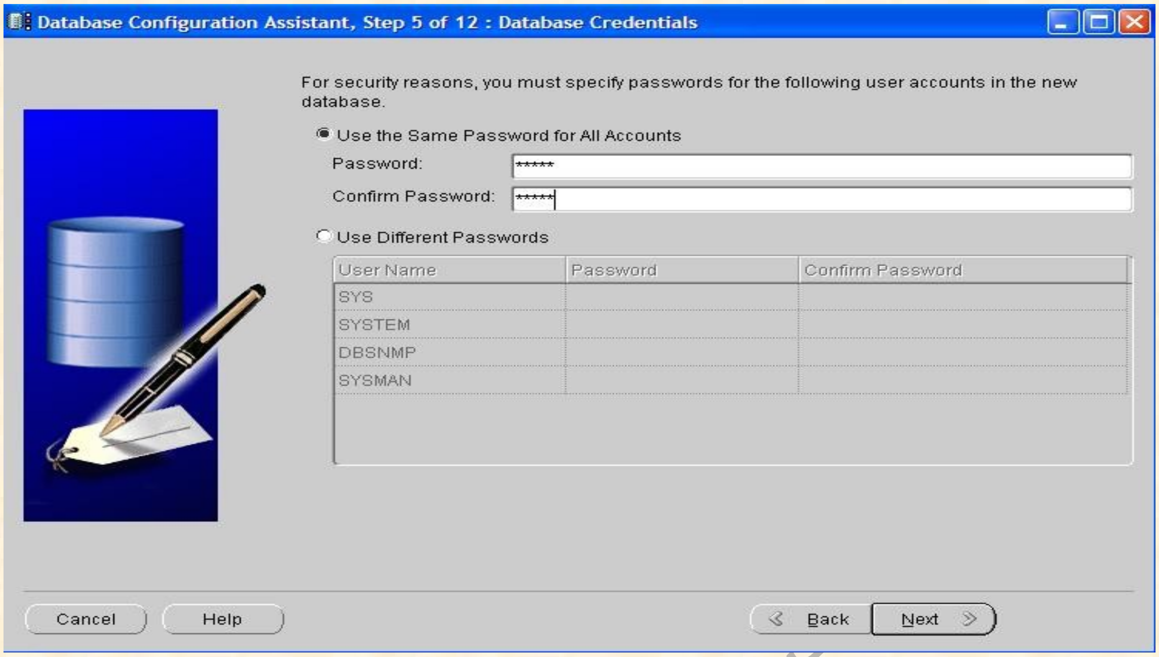
فى هذه الخطوة نختار القالب (Template) الذى يتناسب مع مع متطلباتنا إذا لم يوجد نستطيع إنشاء قالب بالخيار الرابع (Manage Template) .
ولنفترض أننا اخترنا القالب General Purpose .



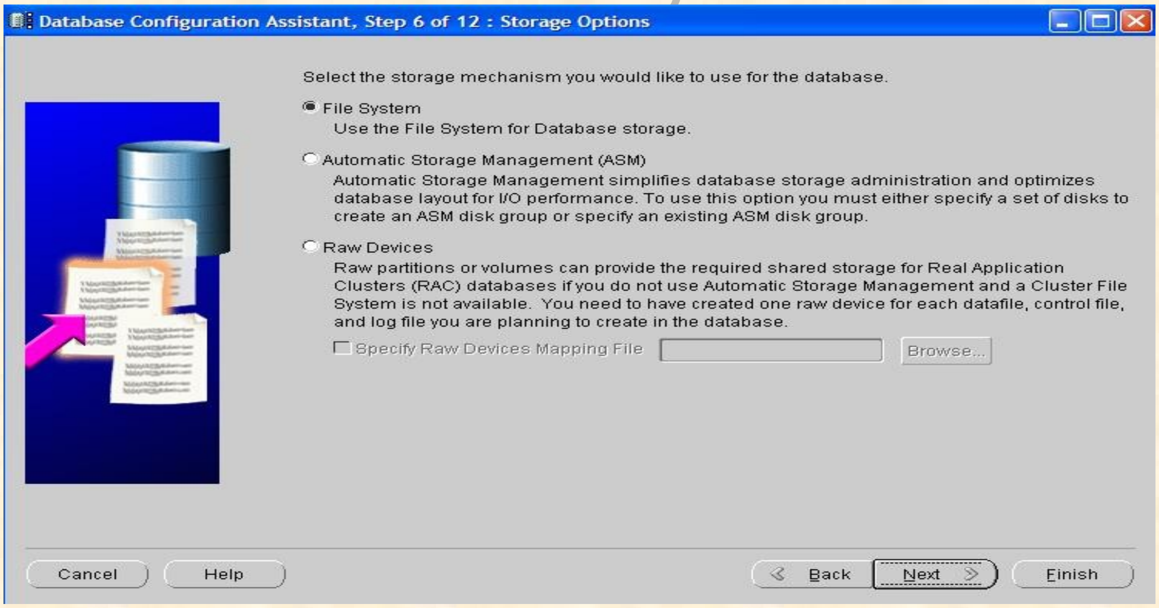
نختار هنا اسم قاعدة البيانات واسم ال(Instance (SID وليس بالضرورى أن يكون نفس الاسم .



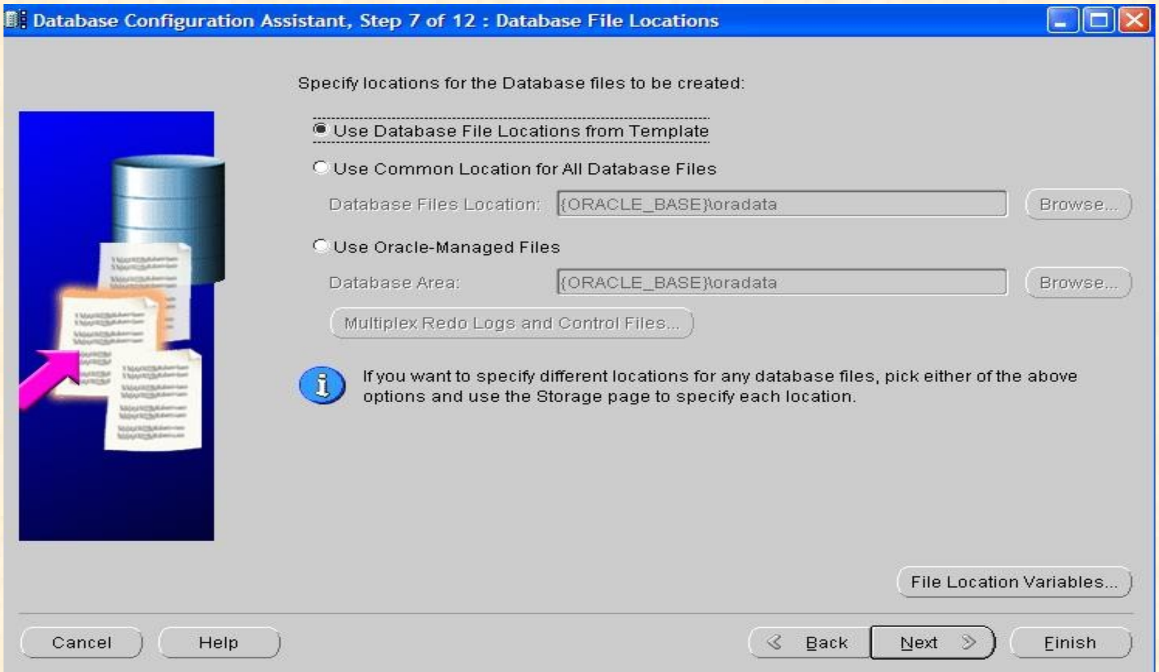
هنا بعض الخيارات كعمل نسخ احتياطى كل فترة معينة او تشغيل الايميل للإرسال والإستقبال .



يمكن عمل كلمة سر واحدة لكل المستخدمين كما فى الخيار الاول ويمكن تحديد كلمة سر لكل مستخدم كما فى الخيار الثانى .

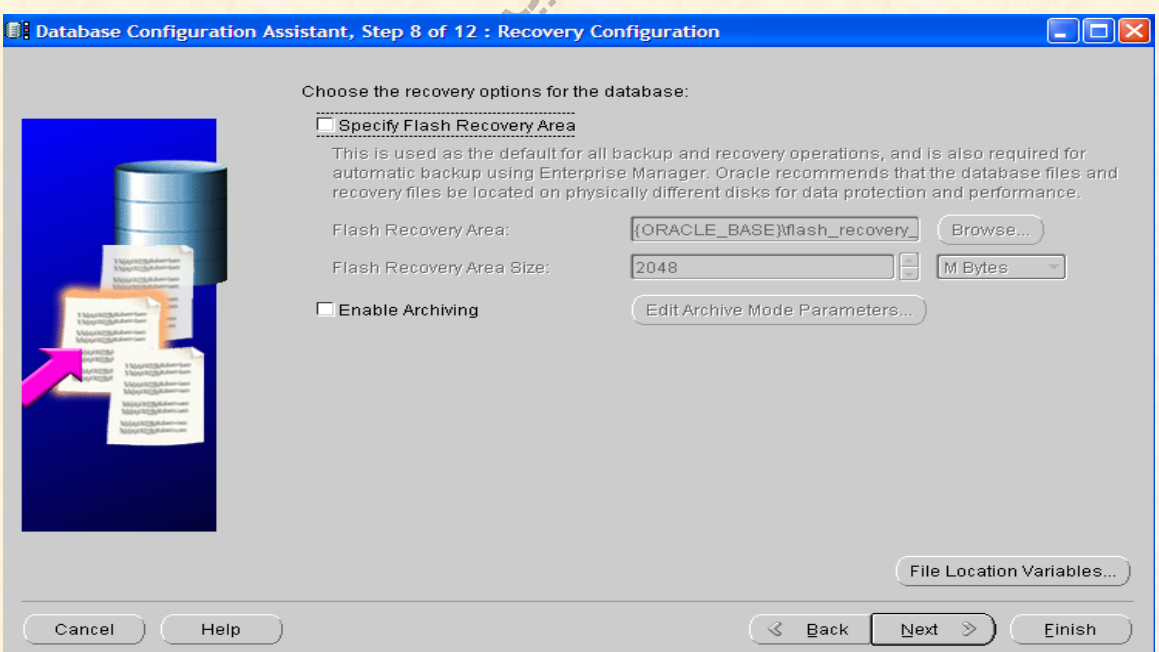


هنا نحدد طريقة تخزين الملفات ،وقد اخترنا الطريقة الاولى وهى ادارة الملفات عن طريق نظام التشغيل.
الطريقة الثانى هى طريقة تخزين وإدارة الية ولها ميزات سنتعرف عليها لاحقاً .



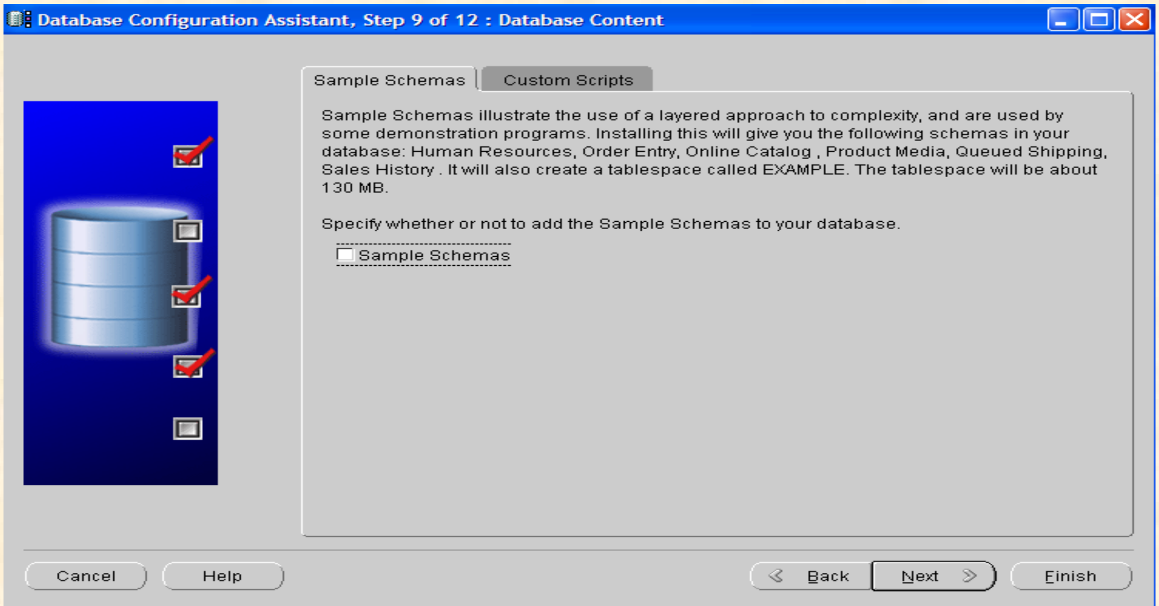
نختار هنا مكان تخزين ملفات قاعدة البيانات . ولنفترض هنا الخيار الاول وهو يعنى التخزين حسب ما هو محدد فى القالب الذى اخترناه والذى كان General Purpose .

يمكن فى هذه الخطوة أن نحدد Flash Recovery Area وهى مكان لتخزين

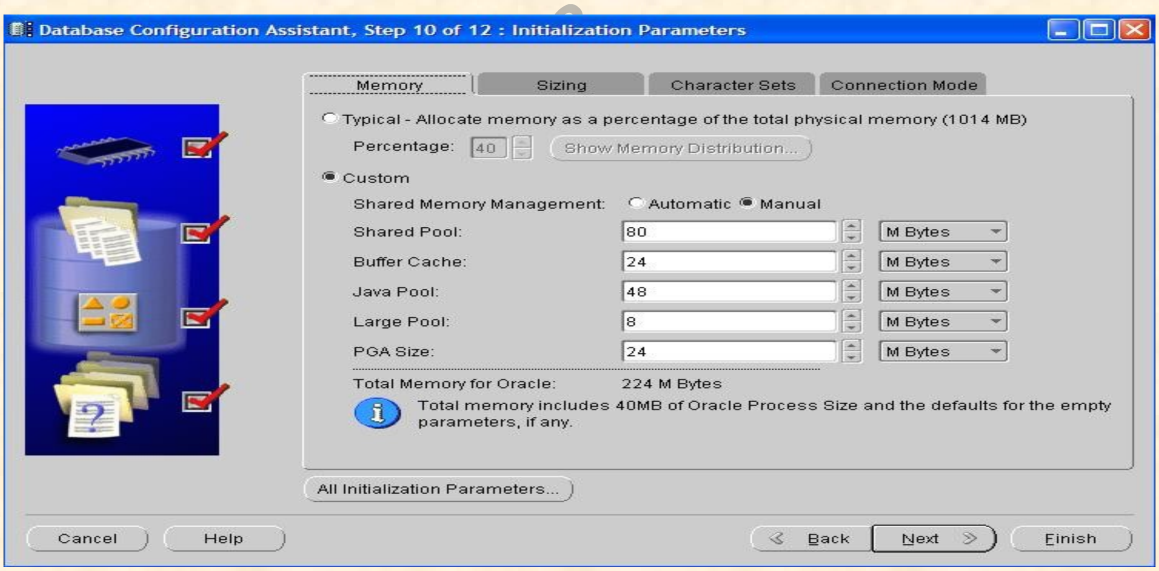


وإدارة عمليات النسخ الاحتياطى والاسترجاع وسنتحدث عنها لاحقاَ , كما يمكن تشغيل الارشيف كما سنعرف لاحقاً .

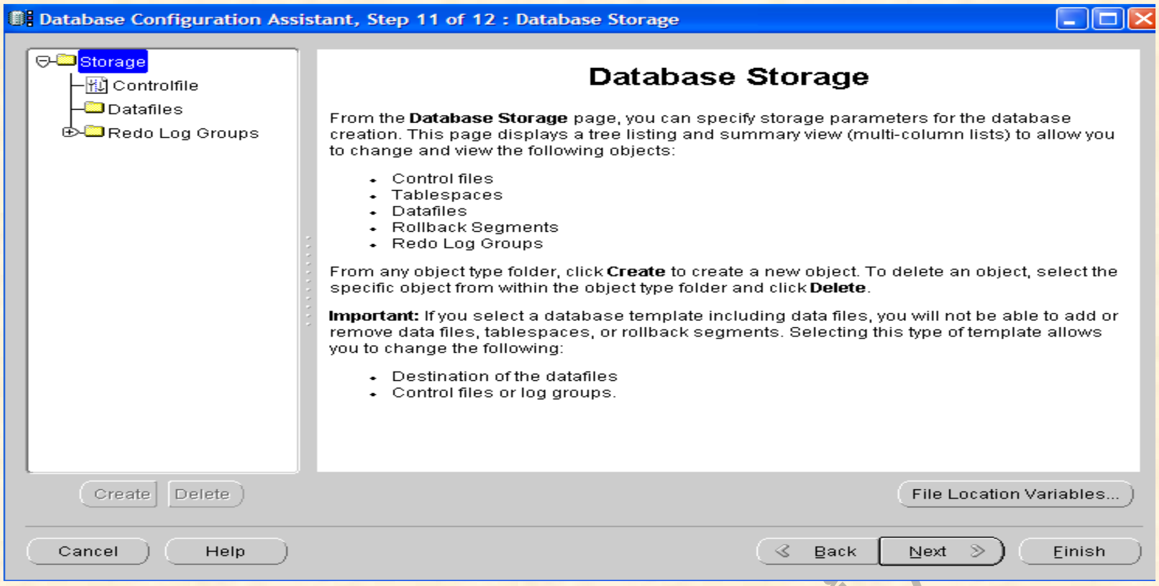
كما يمكن مشاهدة بعض المتغيرات والمعلومات عن قاعدة البيانات بالذهاب ال File Location Variables .



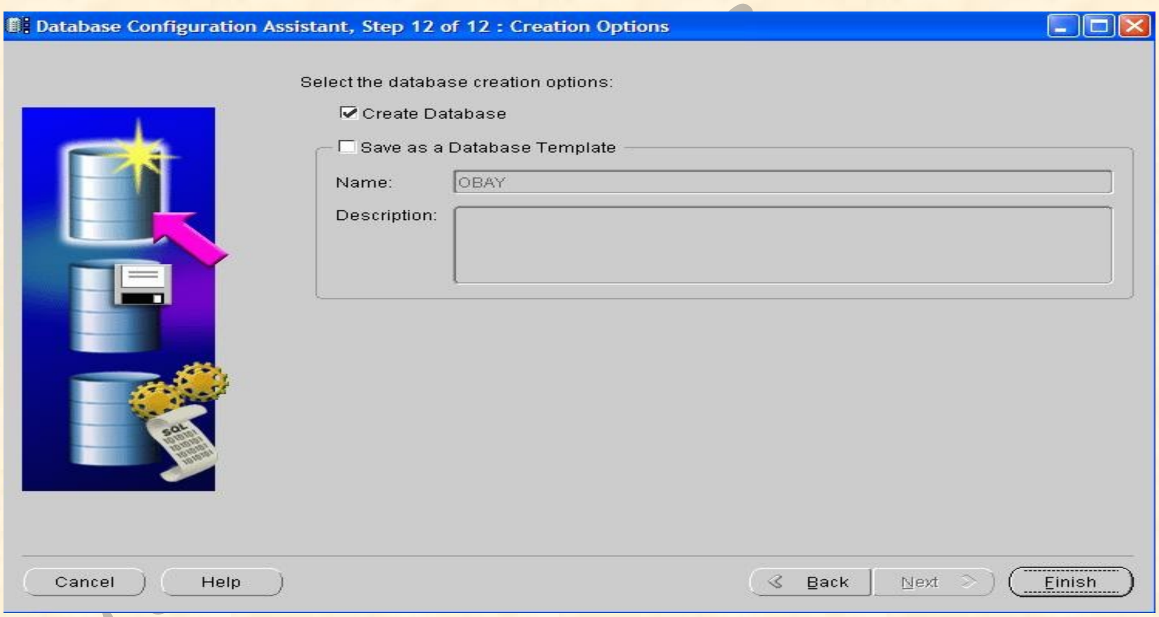
هنا يمكن إنشاء نموزج للSchema للتدريب .



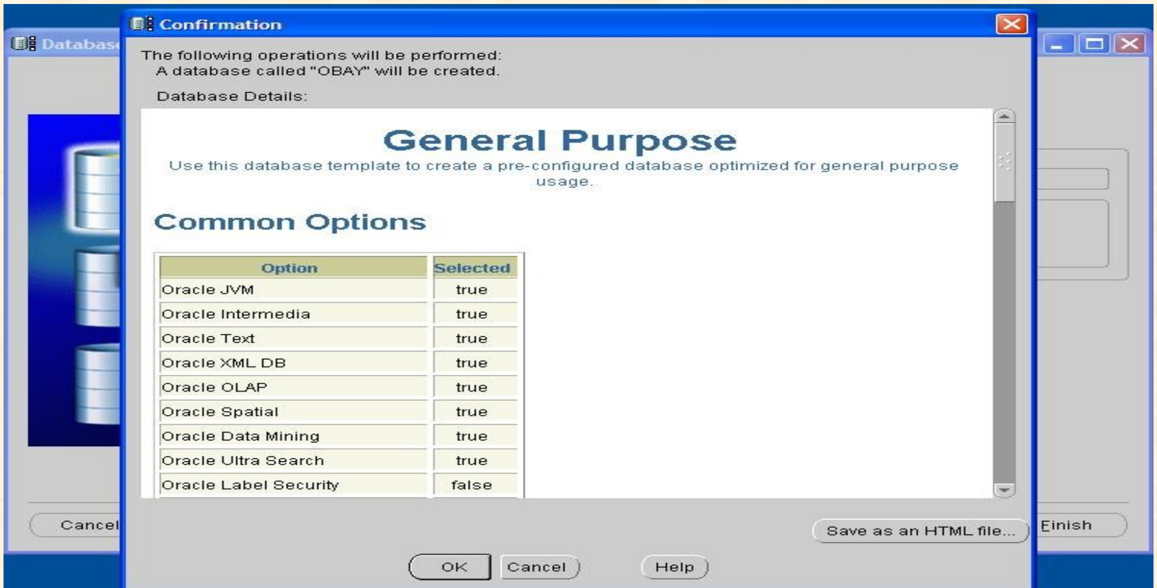
فى هذه الشاشة يمكن أن نغير فى حجم الذاكرة ونحدد حجم الBlock لقاعدة البيانات وغيرها من المتغيرات الموجودة فى ملف المتغيرات .



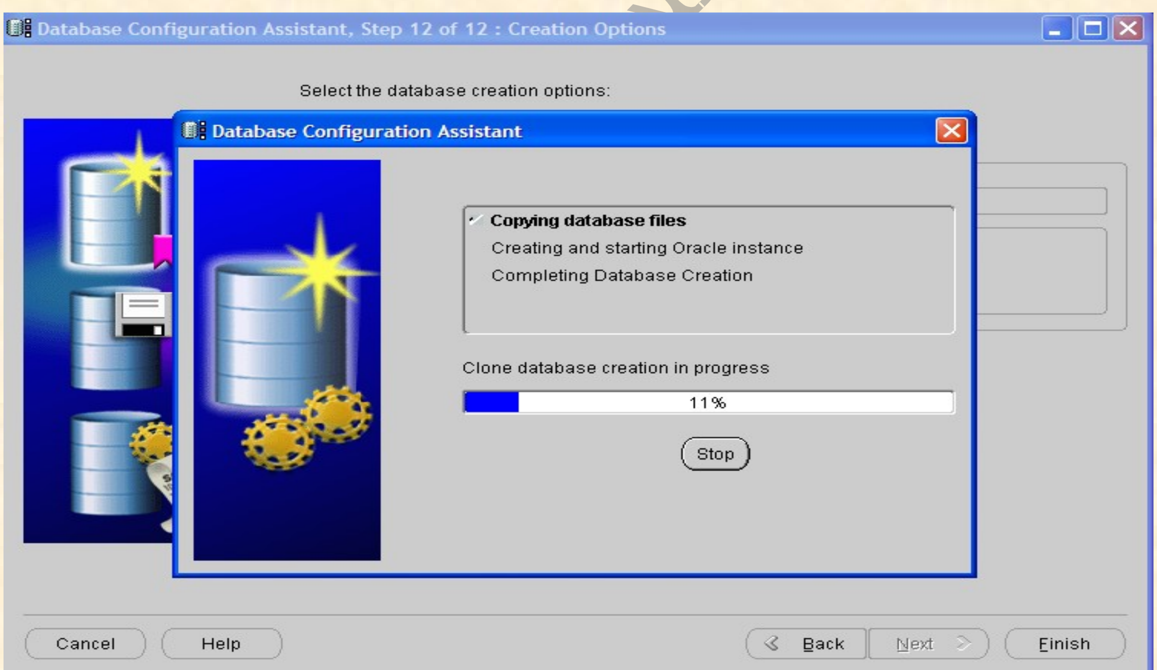
هنا يمكن مشاهدة اماكن تخزين ملفات قاعدة البيانات .



نختار هنا الخيار الاول لانشاء قاعدة البيانات ، كما يمكن حفظ خطوات انشاء قاعدة البيانات كقالب وذلك باختيار الخيار الثانى .
عند الضغط على الخيار Finish تظهر الصفحة التالية :-



وهى عبار عن تفاصيل ومعلومات عن قاعدة البينات التى نريد إنشاءها بالطبع يمكن حفظها بالضغط على الخيار حفظ .



انتظر قليلاً حتى ينتهى أنشاء قاعدة البيانات .

الفصل الثالث

التحكم فى قاعدة البيانات

Controlling the Database

فى الباب السابق قمنا بإنشاء قاعدة البيانات وسنناقش فى هذا الفصل كيفية التحكم فى قاعدة البيانات ، وهو امر مهم لمدير قاعدة البيانات .

تخيل أنك مدير لقاعدة بيانات عملاقة ، يتتطلب منك الامرأن تكون هذه القاعدة متاحة للجميع حتى زمن محدد بعد هذا الوقت تكون متاحة لعدد معين للمستخدمين ، هذا العدد من المستخدمين بعد زمن اخر يستطيع فقط القراءة أى لا يستطيع الكتابة والتعديل ولا المسح .

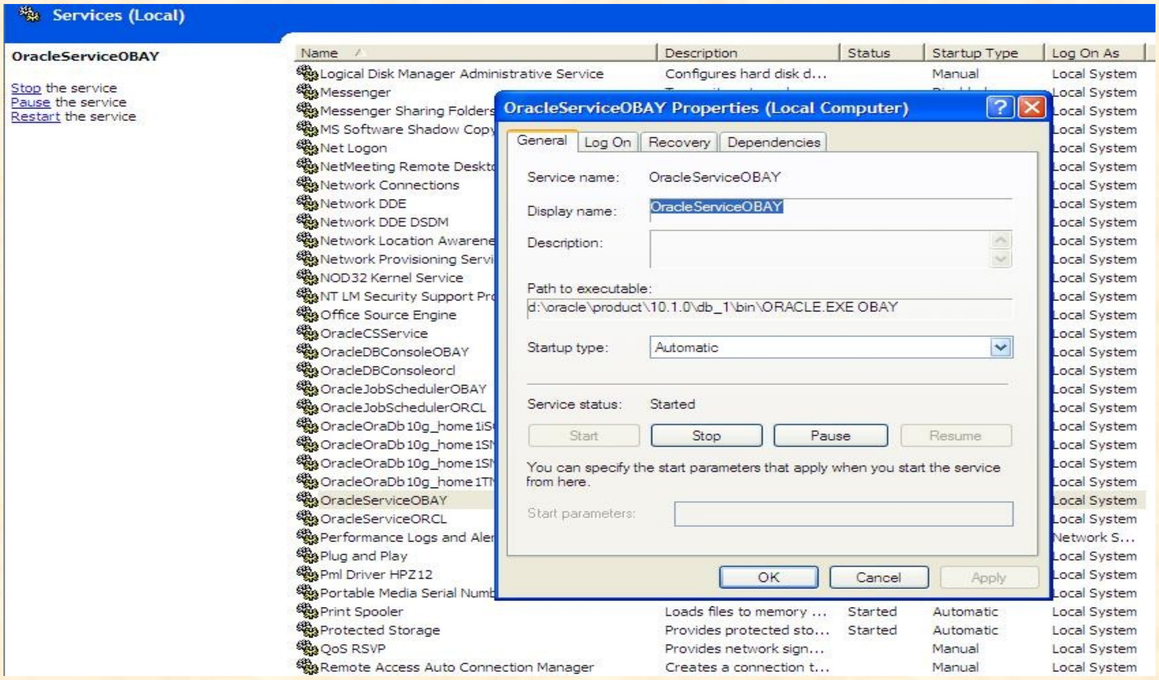
هل تخيلت معى العمل الذى ينتظرك ؛ بالطبع عملاً كبيراً لكن ليس صعباً إذا عرفت ماذا تريد أن تفعل بالضبط .

والان اول ما يجب أن تعرفه أنك تستطيع أن تتحكم فى قاعدة البيانات إذا كانت تعمل فى نظام التشغيل ويندوز عن طريق الServices وهناك عدة خيارات أولاً STOP لإيقاف العمل وكذلك START للتشغيل وايضاً RESTART لإعادة التشغيل وكذلك هنا خيارات اخرى مثلاً (Manual Disabled & & Automatic).

Automatic: والمعنى أنه لحظة تشغيل نظام التشغيل تعمل الServices الياً إذا كانت فى الخيار START.

Manual: وهى أن تقوم بعمل تشغيل يدوى للServices بعد تشغيل نظام التشغيل.

Disabled: والمعنى إيقاف عمل الServices مهما كان حالتها.



ملف المتغيرات (Initialization Parameter File):

وهذا الملف شأنه عجيب إذ لا تعمل قاعدة البيانات دون هذا الملف لذا كان لزاماً أن نتحدث عنه فى هذا الفصل فهو اول ملف تحتاجه قاعدة البيانات عند تشغيلها ، فهو يحتوى على إسم قاعدة البيانات وكذلك اسم ومكان الControl Files وايضاً عن طريقه تتهيأ الذاكرة (SGA) .

فلحظة تشغيل قاعدة البيانات يتم قراءة هذا الملف فيتم تكوين الذاكرة ويتم معرفة اسم ومكان ملف الControl Files.

وهذا الملف قد يكون :-

1. **Static Parameter File (PFILE):** ويكون اسمه (initSID.ora) ، حيث الSID هو اسم الInstance.

وهو ملفى نصى نستطيع أن نجرى عليه التغيرات التى نحتاجها ثم نحفظه وذلك عن طريق نظام التشغيل ، ولكى يحدث التاثير فى قاعدة البيانات لابد من إغلاقها وفتحها من جديد .

وهذا نموزج لملف PFILE :

PFILE Example

```
# Initialization Parameter File: initdb01.ora
db_name                   = db01
instance_name             = db01
control_files             = ( /u03/oradata/db01/control01db01.ctl,
                              /u03/oradata/db01/control02db01.ctl)
db_block_size             = 4096
db_block_buffers          = 500
shared_pool_size          = 31457280  # 30M Shared Pool
db_files                  = 1024
max_dump_file_size        = 10240
background_dump_dest      = /u05/oracle9i/admin/db01/bdump
user_dump_dest            = /u05/oracle9i/admin/db01/udump
core_dump_dest            = /u05/oracle9i/admin/db01/cdump
undo_management           = auto
undo_tablespace           = undtbs
. . .
```

2. **Persistent Parameter File (SPFILE):** يكو ن اسمه spfileSID.ora ، حيث الSID هو اسم الInstance ، وهو ملف ثنائى لا نستطيع التغيير فيه إلى عن طريق الاوركل بواسطة الامر :-

```
ALTER SYSTEM SET %PARAMETER%=VALUE
```

```
ALTER SYSTEM SET undo_tablespace = UNDO;
```

وقد لا نحتاج لإعادة تشغي قاعدة البيانات حتى تحدث التأثيرات وذلك حسب العامل SCOPE .

```
ALTER SYSTEM SET parameter = value [SCOPE =
MEMORY|SPFILE|BOTH]
```

وقد يأخذ العامل SCOPEاحدى ثلاث قيم :-

1- **MEMORY**: وهى تعنى أن التغييرات تحدث فقط فى الInstance التى تعمل الان فأول إعاد ة تشغيل لقاعدة البيانات نفقد التغييرات.

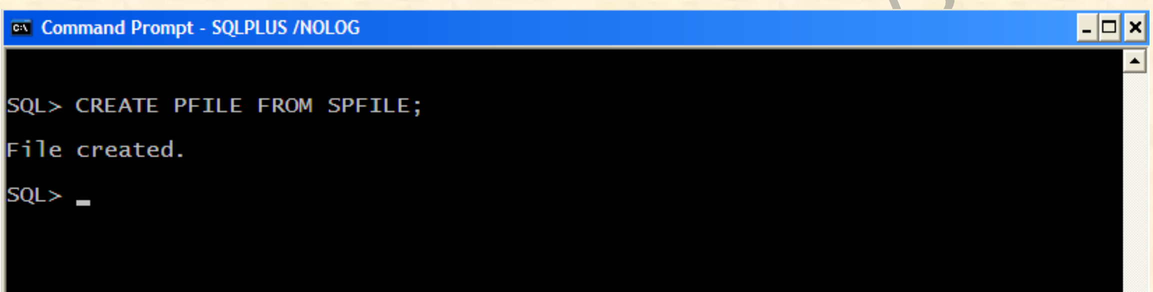
2- **SPFILE**: التغييرات هنا تحدث فى الملف ويحدث التاثير عند إعادة تشغيل قاعدة البيانات.

3- **BOTH**: التغييرات هنا تحدث في الـInstance الحالية كما تحدث أيضا فى الملف SPFILE اى أن التغييرات تظل باقية عند إعادة تشغيل قاعدة البيانات.

والاصل ان التغيرات تحدث فى كل من الـInstance الحالية والـSPFILE اى (BOTH) ولكن يعتمد التغيير ايضاً على المتغير فبعض المتغيرات لا يمكن تعديلها إلا بوسطة الخيار SPIFLE ، اى لا يمكن تغييرها إلا بعد إغلاق قاعدة البيانات.

يمكن إنشاء ملف الـSPFILE من الملف PFILE ولكن يجب أن يملك المستخدم الصلاحية SYSDBA.

```
CREATE SPFILE FROM PFILE;
```



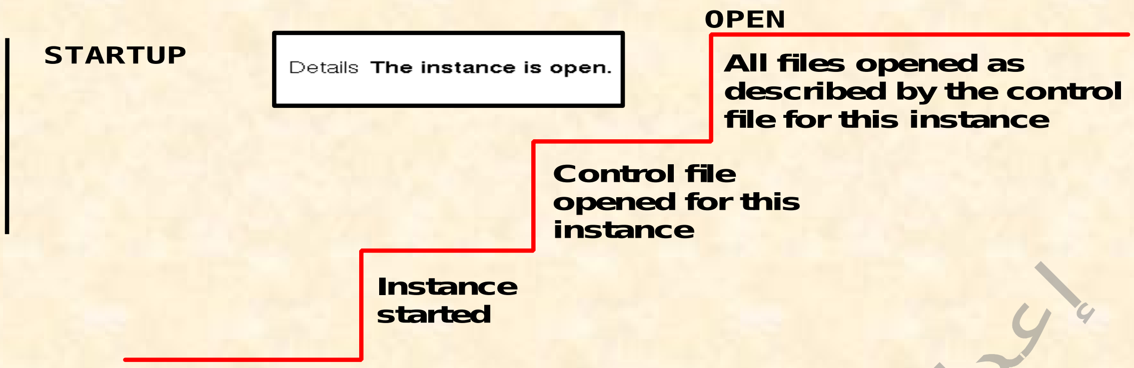
كما يمكن إنشاء ملف الـPFILE من الملف SPFILE.

تشغيل قاعدة البيانات (Starting Up Database):

لتشغيل قاعدة البيانات يلزمك تحديد الحالة التى تريد أن تعمل بها قاعدة بياناتك :-

1- NOMOUNT.
2- MOUNT.
3- OPEN.

Starting Up an Oracle DB Instance:



:NOMOUNT

نشغل الInstance فى هذه الحالة إذا أردنا أن نقوم بإنشاء قاعدة بيانات أو لإعادة إنشاء ملف الControl Files ، وعند تشغيل الInstance فى هذه الحالة تحدث الخطوات التالية :-

1- قراءة ملف المتغيرات وذلك حسب الترتيب التالى :-

* اولاً spfileSID.ora.
* اذا لم يجده يبحث عن spfile.ora.
* إذا لم يجده يبحث عن initSID.ora.

2 - تكوين الSGA.

3 - تشغيل الbackground processes.

4 - فتح ملف الalertSID.log وملف الtrace files .

```
Command Prompt - SQLPLUS /NOLOG
SQL> STARTUP NOMOUNT
ORACLE instance started.

Total System Global Area   171966464 bytes
Fixed Size                    787988 bytes
Variable Size              145488364 bytes
Database Buffers            25165824 bytes
Redo Buffers                  524288 bytes
SQL>
```

:MOUNT

تشغل الInstance فى هذه الحالة عند إجراء بعض العمليات على قاعدة البيانات كتغيير اسم الData files أو انجاز استرجاع كلى لقاعدة البيانات

او تشغيل قاعدة البيانات فى ب عض الاوضاع كوضع الارشيف مثلاُ ، وعند تشغيل الInstance فى هذه الحالة تحدث الخطوات التالية :-

1- الوصول الى ملف الControl files وفتحه بعدما تم تحديده بواسطة ملف المتغيرات .

2- قراءة ملف الControl files لتحديد ملفات الData Files ومعرفة حالتها وكذلك لتحديد ملفات الRedo log Files .

```
Command Prompt - SQLPLUS /NOLOG
SQL> STARTUP MOUNT;
ORACLE instance started.

Total System Global Area  171966464 bytes
Fixed Size                   787988 bytes
Variable Size             145488364 bytes
Database Buffers           25165824 bytes
Redo Buffers                 524288 bytes
Database mounted.
SQL>
```

:OPEN

هذا هو الوضع الاصلى عند تشغيل قاعدة البيانات بحيث تكون الInstance مفتوحة وقاعدة البيانات متاحة بحيث يستطيع المستخدمين الإتصال بقاعدة البيانات وتنفيذ عملياتهم .

واثناء التشغيل فى هذه الحالة تحدث الخطوات التالية :-

1- فتح ملفات الOnline Data Files.

2- فتح ملفات الOnline Redo Log Files.

```
Command Prompt - SQLPLUS /NOLOG
SQL> startup open
ORACLE instance started.

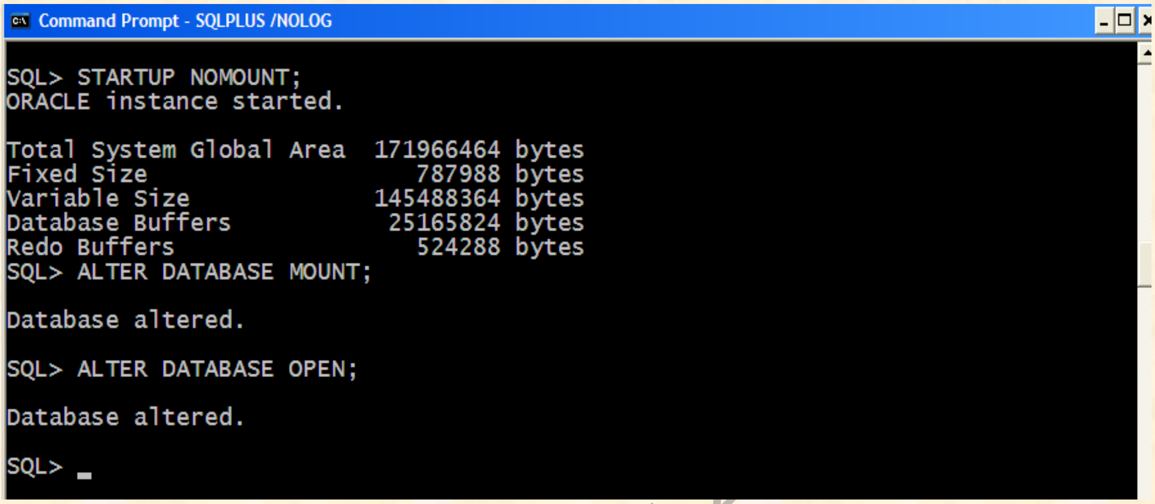
Total System Global Area  171966464 bytes
Fixed Size                   787988 bytes
Variable Size             145488364 bytes
Database Buffers           25165824 bytes
Redo Buffers                 524288 bytes
Database mounted.
Database opened.
SQL>
```

أو

```
Command Prompt - SQLPLUS /NOLOG
SQL> startup
ORACLE instance started.

Total System Global Area  171966464 bytes
Fixed Size                   787988 bytes
Variable Size             145488364 bytes
Database Buffers           25165824 bytes
Redo Buffers                 524288 bytes
Database mounted.
Database opened.
SQL> _
```

هكذا تكون قاعدة البيانات متاحة للمستخدمين ، وإذا كانت قاعدة البيانات مفتوحة مثلاً فى الوضع NOMOUNT OR MOUNT وأردت أن تفتحها فى الوضع OPEN يلزمك استخدام الامر ALTER.



إغلاق قاعدة البيانات (Shutting Down The Database):

عند إغلاق قاعدة البيانات يجب أن تملك الصلاحية SYSDBA OR SYSOPER وهناك عدة اوضاع لإغلاق قاعدة البيانات :-

1- NORMAL.
2- TRANSACTIONAL.
3- IMMEDIATE.
4- ABORT.

Shutdown Mode	A	I	T	N
Allow new connections	No	No	No	No
Wait until current sessions end	No	No	No	Yes
Wait until current transactions end	No	No	Yes	Yes
Force a checkpoint and close files	No	Yes	Yes	Yes

1- NORMAL:

وهو الوضع الاصلى لإغلاق قاعدة البيانات واثناء الإغلاق تحدث الخطوات التالية :-

- لا يسمح بإتصال مستخدم جديد بقاعدة البيانات .
- Oracle Server ينتظر كل المستخدمين من انهاء اتصالهم قبل اكمال إغلاق قاعدة البيانات.
- كل البيانات الموجود فى الBuffer يجب أن تكتب فى الديسك.
- إنهاء الBackground Processes وكذلك إلغاء ال SGAمن الذاكرة.
- إغلاق قاعدة البيانات.
- يتم إغلاق ملفات قاعدة البيانات قبل أغلاق ال Instance.
- لا تحتاج لعملية إسترجاع للInstanceعند فتح قاعدة البيانات من جديد.


```
Command Prompt - SQLPLUS /NOLOG
SQL> SHUTDOWN;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL>
```

TRANSACTIONAL -2 :

فى هذا الوضع من الإغلاق تحدث الخطوات التالية :-

- كل المستخدمين لا يستطيعون بدء عمليات جديدة .
- عند إنتهاء المستخدم من العملية (Transaction) الحالية يتم قطع إتصاله بقاعدة البيانات.
- لحظة إنتهاء كل العمليات (Transactions) فى قاعدة البيانات يتم إغلاق قاعدة البيانات .
- لا تحتاج لعملية إسترجاع للInstanceعند فتح قاعدة البيانات من جديد.

```
Command Prompt - SQLPLUS /NOLOG
SQL> SHUTDOWN TRANSACTIONAL;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> _
```

IMMEDIATE -3 :

فى هذا الوضع من الإغلاق تحدث الخطوات التالية :-

- العمليات الحالية فى قاعدة البيانات يتم قطعها مباشرةً .
- Oracle Server لا ينتظر المستخدميين الحاليين فى قاعدة البيانات حتى ينهوا إتصالهم.
- Oracle Server يقوم بعمل تراجع للعمليات النشطة حالياً فلى ال Instance.
- يتم إغلاق ملفات قاعدة البيانات قبل أغلاق الInstance.
- لا تحتاج لعملية إسترجاع للInstanceعند فتح قاعدة البيانات من جديد .

```
Command Prompt - SQLPLUS /NOLOG
SQL> SHUTDOWN IMMEDIATE;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL>
```

ABORT -4 :

فى هذا الوضع من الإغلاق تحدث الخطوات التالية :-

- العمليات الحالية فى قاعدة البيانات يتم قطعها مباشرةً .
- لا ينتظر المستخدميين الحاليين فى قاعدة البيانات حتى ينهوا إتصالهم.
- البيانات الموجودة فى الBuffer لا تكتب فى الديسك.

- العمليات التى لم يتم تثبيتها لا يتم التراجع عنها .
- يتم إنهاء الInstance دون إغلاق ملفات قاعدة البيانات.
- عند فتح قاعدة البيانات من جديد نحتاج لعمل إسترجاع للInstance.

```
SQL> SHUTDOWN ABORT;
ORACLE instance shut down.
SQL>
```


Opening a Database in Read-Only Mode:

يمكن فتح قاعدة البيانات فى الوضع Read Only وذلك لعدم اجراء أى تعديلات على قاعدة البيانات .
اثناء فتح قاعدة البيانات فى الوضع Read Only يمكن القيام ببعض المهام :-

1- تنفيذ إستعلام .
2- وضع ملفات الData Files فى الوضع offline and online .
3- إنجاز إسترجاع لملفات offline data file and tablespace.

```
SQL> STARTUP OPEN READ ONLY;
ORACLE instance started.

Total System Global Area  171966464 bytes
Fixed Size                   787988 bytes
Variable Size             145488364 bytes
Database Buffers           25165824 bytes
Redo Buffers                 524288 bytes
Database mounted.
Database opened.
SQL>
```

Opening a Database in Restricted Mode:

فى هذا الوضع يمنع دخول المستخدمين لقاعدة البيانات إلا إذا كان المستخدم يملك صلاحية RESTRICTED SESSION PRIVILEGE.

فى الحقيقة هذا الوضع مفيد فى البعض الاحيان مثلاً اثناء قيامك بعمل تصدير Export لقاعدة البيانات ولا تريد من المستخدمين من الدخول لقاعدة البيانات .

```
Command Prompt - SQLPLUS /NOLOG
SQL> STARTUP RESTRICT;
ORACLE instance started.

Total System Global Area  171966464 bytes
Fixed Size                   787988 bytes
Variable Size             145488364 bytes
Database Buffers           25165824 bytes
Redo Buffers                 524288 bytes
Database mounted.
Database opened.
SQL>
SQL>
```

او

الان كل المستخدمين الذين لا يملكون صلاحية ال RESTRICTED SESSION

```
Command Prompt - SQLPLUS /NOLOG
SQL> ALTER SYSTEM ENABLE RESTRICTED SESSION;

System altered.

SQL>
SQL>
```

PRIVILEGE لا يستطيعون دخول قاعدة البيانات .

الان يمكن إنهاء كل اتصالات المستخدمين الذين اتصلوا قبل تطبيق النمط RESTRICTED SESSION وذلك على خطوتين :-

1- تحديد الSESSION التى نريد إلغاءها وذلك عن طريق الإستعلام الاتى :

```
SELECT SID, SERIAL# FROM V$SESSION
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> SELECT SID,SERIAL# FROM V$SESSION;

       SID    SERIAL#
---------- ----------
       153          5
       159          1
       160          9
       161         15
       162          3
       163          1
       164          1
       165          1
       166          1
       167          1
       168          1

       SID    SERIAL#
---------- ----------
       169          1
       170          1

13 rows selected.

SQL> _
```

2- إلغاء الSESSION.

```
Command Prompt - SQLPLUS /NOLOG

SQL> ALTER SYSTEM KILL SESSION '153,5';

System altered.

SQL>
```

الفصل الرابع
Managing Tablespaces and Data files

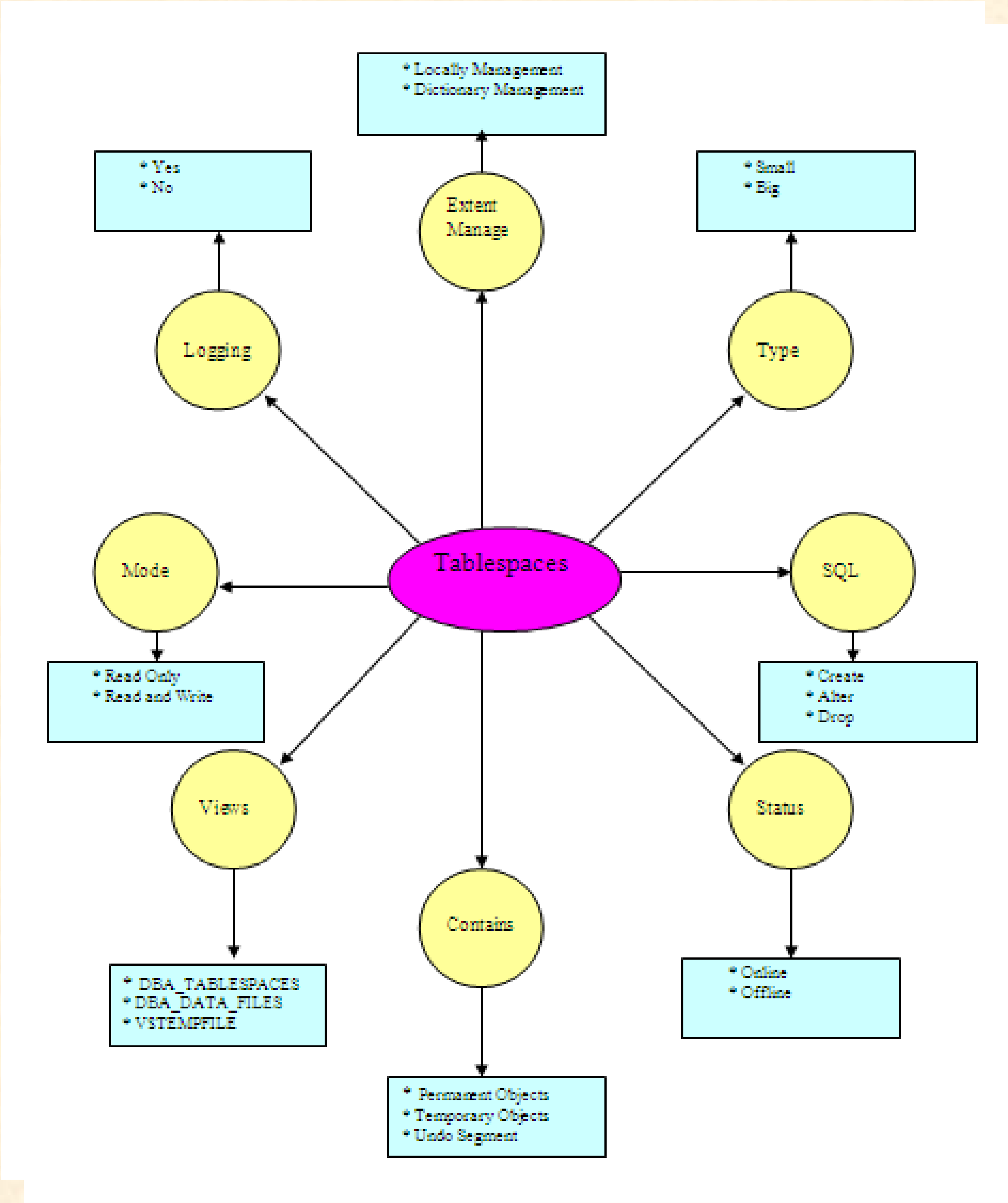
تحدثنا سابقاً ان التخزين فى قاعدة البيانات يكون على حالتين الاول التخزين المنطقى Logical فى الTablespaces والثانى Physical فى ال Data Files .

وذكرنا كذلك أن اى Data Files يكون فى قاعدة البيانات هو فى الحقيقة ينتمى لTablespace واحد .

بمعنى اخر أنه لا يوجد Data Files فى قاعدة البيانات لا ينتمى ل Tablespace .

وعموماً فى هذا الفصل سنتحدث عن :-

1- مفاهيم عامة عن الTablespaces.
2- إنشاء الTablespaces.
3- إدارة الTablespaces.
4- كيفية الحصول على معلومات عن الTablespaces.



1- مفاهيم عامة عن الTablespaces:

الشكل اعلاه يوضح الخيارات المتاحة للTablespaces وهى :

1 - Space Management in Tablespaces: وهى كيفية إدارة المساحة فى الTablespaces وهناك نوعان:-

1- <u>Locally Managed Tablespaces</u>.

هنا يتم إدارة الExtents فى الTablespace عن طريق ال Tablespace بواسطة ال Bitmaps، فلحظة تخصيص الExtents أو تحريرها يقوم الOracle Server بتغير قيمة الBitmap للحالة الجديدة . وهذا النوع من الإدارة هو الاصل عند إنشاء الTablespace فى الاصدار Oracle 10g ، وقد صار هذا النوع متاح إبتداء من الإصدار Oracle 8i , ولانه لا يحدث تعديل فى الDictionary Data فإنه لا يتم Generate Undo Information.

لتحويل إدارة الTablespace من ال Data Dictionaryالى Locally نستخدم

DBMS_SPACE_ADMIN.TABLESPACE_MIGRATE_TO_LOCAL

فى هذا النوع من إدارة الTablespace هناك نوعان لإدارة ال Extents:

1- Automatic: هنا لا نحدد حجم الExtent إنما يتم تحديده عن طريق النظام ، ولا يمكن تطبيق هذا النوع فى (Temporary Tablespace).

2- Uniform: هنا يمكن تحديد حجم الExtent فى ال Tablespace والاصل حجم الextent هو IMB. لا يمكن تطبيق هذا النوع فى Undo Tablespace.

وكذلك فى هذا النوع من إدارة الTablespace هناك نوعان لإدارة ال Segement:

1- Automatic.

2- Manual.

2- Dictionary- Managed Tablespaces:

هنا يتم إدارة الExtents فى الTablespace عن طريق الData Dictionary

فلحظة تخصيص الExtents أو تحريرها يقوم الOracle Server بتغير الData Dictionary Tables .

2- Logging:

1- **<u>Yes</u>**: المعنى لحظة تغير الكائنات فى الTablespaces فانه يتم كتابة التغييرات فى الRedo Log Files.

2- **NO**: المعنى لحظة تغير الكائنات فى الTablespaces فانه لا يتم كتابة كل التغييرات فى الRedo Log Files.

3- Mode:

1- Read Only: المعنى هنا أننا لا نستطيع الكتابة أو التعديل أو المسح على البيانات الموجودة فى الTablespace ، بالطبع لا نستطيع أن نضع كل من (الSystem & Sysaux Tablspace) فى الوضع Read Only.

2- Read and write: المعنى أننا نستطيع الكتابة أو التعديل أو المسح على البيانات الموجودة فى الTablespace.

4- Views:

وهى الإستعلامات لمعرفة معلومات عن الTablespaces والData Files..

DBA_TABLESPACES
V$TABLESPACE
DBA_DATA_FILES
V$DATAFILE
DBA_TEMP_FILES
V$TEMPFILE

5- Contains:

1- Permanent : وهو لتخزين الكائنات المستديمة فى قاعدة البيانات .

2- Temporary : لتخزين الكائنات المؤقتة فى قاعدة البيانات لذلك تحتاج قاعدة البيانات ، مثلاً لعمليات ترتيب البيانات.

3- Undo : تستخدمه الDatabase Server لتخزين الUndo Information وهو ضرورى فى قاعدة البيانات ويتم إنشاءه أثناء إنشاء قاعدة البيانات ويمكن اضافة اخرى عندما نحتاج لذلك لكن فى الوقت الواحد يعمل Undo واحد فقط يحدد عن طريق ملف المتغيرات %Undo_Tablespace=%Value. .

.

6- Status:

1- Online: أى أن **ال**Tablespace متاح لجميع المستخدمين .

2- Offline : اى أن الTablespace غير متاح للمستخدمين وهناك عدة خيارات لهذا الوضع (Normal & Temporary & Immediate & For Recover) . لا يمكن وضع (System & Undo Tablespace) فى الوضع Offline.

7- SQL:

1- Create : لإنشاء الTablespace.

2- Alter: لإجراء تغيرات على الTablespace.
3- Drop: لحذف الTablespace.

8- Type:
1- Small Tablespace: وهو Tablespace يحتوى على Data File واحد أو أكثر وهو الاصل عند إنشاء Tablespace.
2- Big Tablespace: وهو نوع جديد من الTablespace صار متاح فى الإصدار Oracle 10g ، بحيث تصل مساحته الى 128 Terabytes(TB) ويحتوى على ملف data File واحد فقط .

تعريف عن بعض الTablespace:

1- System Tablespace:
هو أهم Tablespace فى قاعدة البيانات على الإطلاق إذ لا تعمل قاعدة البيانات بدونه كذلك لا نستطيع تغير حالته إلى الوضع (Read only & Offline) .
يستخدم هذا الTablespace من قبل الOracle Server لإدارة قاعدة البيانات ، حيث يحتوى على الData Dictionary والجداول التى تحتوى على معلومات إدارية حول قاعدة البيانات ، كذلك يحتوى على (Sys Schema).

2- Sysaux Tablespace:
وهو Tablespace مساعد للSystem Tablespace يحتوى على بعض المكونات والمنتجات التى تستخدم فى إدارة قاعدة البيانات . وهذا ال Tablespace صار ضرورى فى قاعدة البيانات إبتداءً من الإصدار Oracle10g .

3- Temp:

وهو Tablespace يستخدم لتخزين الجداول والمراجع المؤقتة ، مثلا عند الترتيب . وقد يكون هناك أكثر من Temporary Tablespace فى قاعدة البيانات الواحدة ولكن يعمل واحد فى اللحظة على قاعدة البيانات .

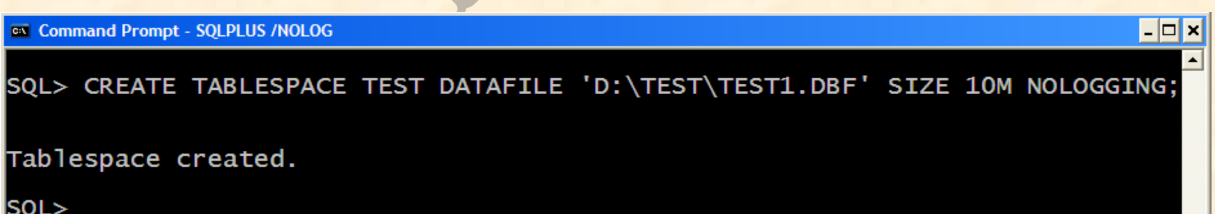
4- Undo:

يستخدم هذا الTablespace من قبل الOracle Server لتخزين الUndo Information
، هل فكرت كيف يمكن التراجع عن عملية تعديل لحقل معين ، المعلومات القديمة يتم حفظها فى هذا الTablespace.

عملى:

1- انشاء Tablespace جديد :

```
CREATE TABLESPACE TEST DATAFILE 'D:\TEST\TEST1DBF'
SIZE 10M NOLOGGING;
```



بما أننا لم نحدد نوع إدارة الTablespace فإن الاصل هو Locally Managed Tablespace.

```
CREATE TABLESPACE TEST DATAFILE 'D:\TEST\TEST.DBF'
SIZE 10M EXTENT MANAGEMENT LOCAL UNIFORM 100K;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> CREATE TABLESPACE TEST DATAFILE 'D:\TEST\TEST.DBF' SIZE 10M EXTENT MANA
GEMENT LOCAL UNIFORM SIZE 100K;

Tablespace created.

SQL>
```

```
CREATE TABLESPACE TEST DATAFILE 'D:\TEST\TEST.DBF'
SIZE 10M EXTENT MANAGEMENT LOCAL AUTOALLOCATE;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> CREATE TABLESPACE TEST DATAFILE 'D:\TEST\TEST.DBF' SIZE 10M EXTENT MANA
GEMENT LOCAL AUTOALLOCATE;

Tablespace created.

SQL>
```

2- زيادة حجم الTablespaces

الطريقة الاولى إضافة Data File جديد للTablespace.

```
ALTER TABLESPACE TEST ADD DATAFILE ‘D:\TEST\
TEST.DBF’ SIZE 5M;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> ALTER TABLESPACE TEST ADD DATAFILE 'D:\TEST\TEST1.DBF' SIZE 5M;

Tablespace altered.

SQL>
```

الطريقة الثانية زيادة حجم الData File الموجودة .

```
ALTER DATABASE DATAFILE 'D:\TEST\TEST.DBF' RESIZE
12M;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> ALTER DATABASE DATAFILE 'D:\TEST\TEST.DBF' RESIZE 12M;

Database altered.

SQL>
```

3- حزف الTablespace:

الطريقة الاولى مسح الTablespace فقط دون الData Files.

```
DROP TABLESPACE TEST;
```

```
Command Prompt - SQLPLUS /NOLOG
SQL> DROP TABLESPACE TEST;
Tablespace dropped.
SQL>
```

بهذه الطريقة يظل الData File موجود على نظام التشغيل ويمكن إستخدامه عند إنشاء Tablespace جديد .

```
CREATE TABLESPACE OBAY DATAFILE 'D:\TEST\TEST.DBF'
REUSE;
```

```
Command Prompt - SQLPLUS /NOLOG
SQL> CREATE TABLESPACE OBAY DATAFILE 'D:\TEST\TEST.DBF' REUSE;
Tablespace created.
SQL>
```

الطريقة الثانية حزف الTablespace ومعه الData File.

```
DROP TABLESPACE OBAY INCLUDING CONTENTS AND
DATAFILES;
```

```
Command Prompt - SQLPLUS /NOLOG
SQL> DROP TABLESPACE OBAY INCLUDING CONTENTS AND DATAFILES;
Tablespace dropped.
SQL>
```

4- تحويل الTablespace إلى الوضع Read Only:

```
ALTER TABLESPACE TEST READ ONLY;
```

```
SQL> ALTER TABLESPACE TEST READ ONLY;

Tablespace altered.

SQL>
```

حاول أن تضع ال System Tablespace فى الوضع Read Only.

```
SQL> ALTER TABLESPACE SYSTEM READ ONLY;
ALTER TABLESPACE SYSTEM READ ONLY
*
ERROR at line 1:
ORA-01643: system tablespace can not be made read only


SQL>
```

لاحظ الرسالة كذلك الحال عن الSysaux Tablespace.

5- تحويل الTablespace الى الوضع Read Write:

```
ALTER TABLESPACE TEST READ WRITE;
```

```
SQL> ALTER TABLESPACE TEST READ WRITE;

Tablespace altered.

SQL>
```

```
SQL> ALTER TABLESPACE TEST OFFLINE;

Tablespace altered.

SQL>
```

6- تحويل الTablespace الى الوضع Offline:

```
ALTER TABLESPACE TEST OFFLINE
```

7 تحويل الTablespace الى الوضع Online:

```
ALTER TABLESPACE TEST OFFLINE;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> ALTER TABLESPACE TEST ONLINE;

Tablespace altered.

SQL>
```

8- لإعادة تسمية الTablespace:

```
ALTER TABLESPACE TEST RENAM TO EXAM;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> ALTER TABLESPACE TEST RENAME TO EXAM;

Tablespace altered.

SQL>
```

9- إنشاء Temporary Tablespace:

```
CREATE TEMPORARY TABLESPACE TEMPEXAM TEMPFILE 'D:\
TEST\TEMPEXAM.DBF' SIZE 7M;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> CREATE TEMPORARY TABLESPACE TEMPEXAM TEMPFILE 'D:\TEST\TEMPEXAM.DBF' SI
ZE 7M;

Tablespace created.

SQL>
```

وحتى يكون هذا الTablespace هو الDefault Temporary Tablespace فى قاعدة البيانات.

```
Command Prompt - SQLPLUS /NOLOG

SQL> ALTER DATABASE DEFAULT TEMPORARY TABLESPACE TEMPEXAM;

Database altered.

SQL>
```

يمكن التأكد من الDefault Temporary Tablespace .

```
SQL> SELECT PROPERTY_NAME,PROPERTY_VALUE FROM DATABASE_PROPERTIES WHERE PROPERTY
_NAME='DEFAULT_TEMP_TABLESPACE';

PROPERTY_NAME
------------------------------
PROPERTY_VALUE
--------------------------------------------------------------------------------

DEFAULT_TEMP_TABLESPACE
TEMPEXAM


SQL>
```

10- إنشاء Temporary Tablespace Group:

وهو ميزة مستحدثة فى الإصدار Oracle 10g تستخدم عند نقص ال Resource المستخدم للتخزين المؤقت .

```
CREATE TEMPORARY TABLESPACE TEMPG1 TEMPFILE 'D:\
TEST\TEMPG1.DBF' SIZE 4M TABLESPACE GROUP GROUB1;
```

```
SQL> CREATE TEMPORARY TABLESPACE TEMPG1 TEMPFILE 'D:\TEST\TEMPG1.DBF' SIZE 4
M TABLESPACE GROUP GROUB1;

Tablespace created.

SQL>
```

أنشاء Tablespace اخر لنفس الGroup.

```
SQL> CREATE TEMPORARY TABLESPACE TEMPG2 TEMPFILE 'D:\TEST\TEMPG2.DBF' SIZE 4
M TABLESPACE GROUP GROUB1;

Tablespace created.

SQL>
```

إضافة Tablespace موجود أصلاً للGroup.

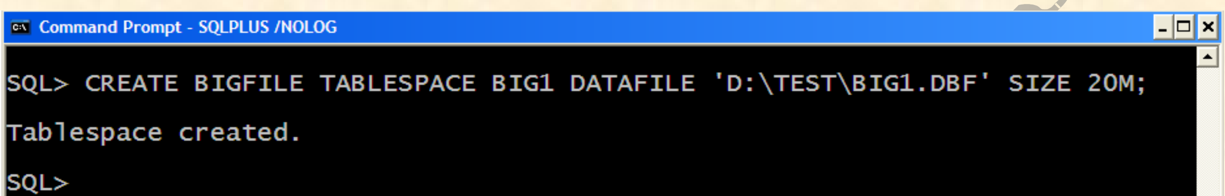
```
ALTER TABLESPACE TEMPEXAM TABLESPACE GROUP GROUP1;
```

```
Tablespace altered.

SQL>
```

11- إنشاء Big Tablespace:

```
CREATE BIGFILE TABLESPACE BIG1 DATAFILE ‘D:\TEST\
BIG1.BDF’ SIZE 20M;
```



للاستعلام عن الTABLESPACES & DATAFILES

DBA_TABLESPACES
USER_TABLESPACES
DBA_TABLESPACE_GROUPS
V$TABLESPACE
DBA_DATA_FILES
DBA_TEMP_FILES

الفصل الخامس
Control File & Redo Log Files

:Control File

او يمكن أن نسميه ملف التحكم وهو ملف ثنائى اى لا يمكن قراءته إذ أنه ليس نصياً ، وبدونه لا تعمل قاعدة البيانات ، وييم قراءته فى الوضع Mount ، إذاً يمكن أن نفتح قاعدة البيانات فى الوضع Nomount حتى لو فقدنا ملف ال Control File .

وبالطبع يكون تابع لقاعدة بيانات واحدة ، ويتم تحديثه فقط عن طريق ال Oracle Server اثناء عمل قاعدة البيانات فلا يستطيع مدير قاعدة البيانات تحديثه .

إذا فقدنا الControl File نحتاج لإسترجاعه قبل فتح قاعدة البيانات من جديد .

يتم إنشاؤه أثناء إنشاء قاعدة البيانات ، كما يمكن زيادة عدد الControl File بعد إنشاء قاعدة البيانات حتى نستطيع الاسترجاع إذا فقدنا احد الControl File .

ويحتوى الControl Files على الاتى :-

1- اسم قاعدة البيانات.

2- وقت إنشاء قاعدة البيانات.
3- اسماء الTablespaces.
4- إسم ومكان الData Files والRedo Log Files.
5- الRedo Log الحلى وكذلك رقمه المتسلسل .
6- ومعلومات عن النسخ الاحتياطى . مثلاً معومات الRMAN . سناقش ذلك لاحقاً .
7- ومعلومات عن الإرشيف .
8- يحتوى على المعلومات التالية (. MAXLOGFILES MAXLOGMEMBERS. MAXLOGHISTORY. MAXDATAFILES .MAXINSTANCES).

:Multiplexing the Control File

والمعنى أن قاعدة البيانات تعمل على عدد من الControl File مع العلم أن كل ملفات الControl File هى طبق الاصل ، نستطيع بهذه الطريقة تأمين قاعدة البينات فى حال فقدان احد الControl File.

والافضل كذلك توزيع هذه الملفات على عدد من الDisk حتى لا نفقدها جميعاً .

الان نفترض أن لدينا قاعدة البيانات اسمها OBAY تحتوى على 3 ملفات من الControl Files نريد أن نضيف لها ملف Control File اخر .

سنقوم بعمل هذا السناريو فى حالين :-

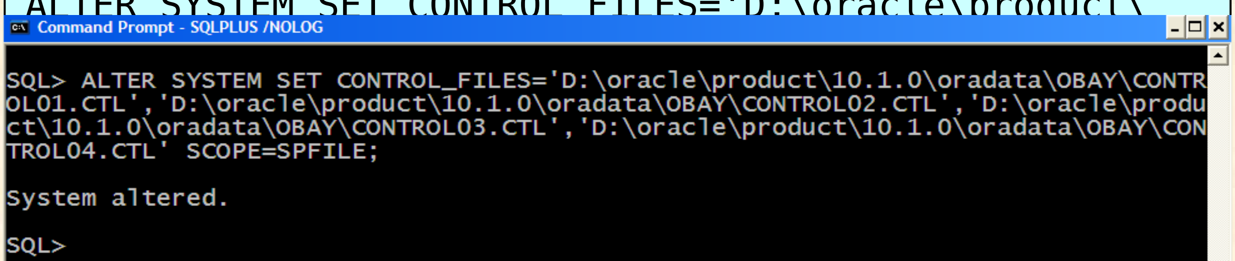
الحال الاول : باستخدام الSPFILE.

الحال الثانى : باستخدام الinit.ora.

الحالة الاولى (SPFILE) :-

1- نقوم بتعديل المتغير Control_files على ملف المتغيرات (SPFILE) ونضيف اليه مكان الControl File الذى نريد أن نضيفه لقاعدة البيانات .

```
ALTER SYSTEM SET CONTROL_FILES='D:\oracle\product\
```



```
SQL> ALTER SYSTEM SET CONTROL_FILES='D:\oracle\product\10.1.0\oradata\OBAY\CONTR
OL01.CTL','D:\oracle\product\10.1.0\oradata\OBAY\CONTROL02.CTL','D:\oracle\produ
ct\10.1.0\oradata\OBAY\CONTROL03.CTL','D:\oracle\product\10.1.0\oradata\OBAY\CON
TROL04.CTL' SCOPE=SPFILE;

System altered.

SQL>
```

2- يلزمنا إغلاق قاعدة البيانات الان حتى يتم تحديث التعديل السابق .

```
SQL> SHUT IMMEDIATE;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL>
```

3- عن طريق نظام التشغيل نقوم بعمل نسخ لاحد ملفات الControl Files ال موجودة الان ونضعه ونسميه كما حددناه فى المتغير Control_Files

```
D:\oracle\product\10.1.0\oradata\OBAY>COPY CONTROL01.CTL CONTROL04.CTL
        1 file(s) copied.

D:\oracle\product\10.1.0\oradata\OBAY>
```

4- الان يمكن فتح قاعدة البيانات .

```
SQL> STARTUP;
ORACLE instance started.

Total System Global Area  171966464 bytes
Fixed Size                   787988 bytes
Variable Size             145488364 bytes
Database Buffers           25165824 bytes
Redo Buffers                 524288 bytes
Database mounted.
Database opened.
SQL>
```

وهكذا قاعدة البيانات الان تعمل باربعة من الControl Files بدلاً من ثلاثة .

```
SQL> SHOW PARAMETERS CONTROL_FILES;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
control_files                        string      D:\ORACLE\PRODUCT\10.1.0\ORADA
                                                 TA\OBAY\CONTROL01.CTL, D:\ORAC
                                                 LE\PRODUCT\10.1.0\ORADATA\OBAY
                                                 \CONTROL02.CTL, D:\ORACLE\PROD
                                                 UCT\10.1.0\ORADATA\OBAY\CONTRO
                                                 L03.CTL, D:\ORACLE\PRODUCT\10.
                                                 1.0\ORADATA\OBAY\CONTROL04.CTL
SQL>
```

لاحظ أن العملية تتطلب إغلاق لقاعدة البيانات ومن ثم عمل نسخ على نظام التشغيل ومن ثم فتح قاعد البيانات من جديد .

الحالة الثانية (init.ora) :-

فى الحالة الاولى قاعدة البيانات كانت تعمل على ملف المتغيرات (SPFILE) والان نفترض أن قاعدة البيانات تعمل على ملف المتغيرات (initobay.ora) :-

1- قفل قاعدة البيانات .

```
Command Prompt - SQLPLUS /NOLOG

SQL> SHUT IMMEDIATE;
Database closed.
Database dismounted.
ORACLE instance shut down.
```

2- تعديل المتغير Control_Files على ملف المتغيرات initobay.ora.

وذلك عن طريق نظام التشغيل .

```
Command Prompt

D:\oracle\product\101~1.0\db_1\database>edit initobay.ora_
```

عند كتابة الامر اعلاه سيفتح ملف المتغيرات ونعدل فى المتغير Control_Files بحيث نكتب المسار الجديد واسم الملف الجديد ، ولنفترض أن اسمه CONTROL05 فى نفس المسار .

```
Command Prompt - edit initobay.ora
   File   Edit   Search   View   Options   Help
            D:\oracle\product\10.1.0\db_1\database\INITobay.ORA
*.background_dump_dest='D:\oracle\product\10.1.0\admin\OBAY\bdump'
*.compatible='10.1.0.2.0'
*.control_files='D:\oracle\product\10.1.0\oradata\OBAY\CONTROL01.CTL','D:\orac
*.core_dump_dest='D:\oracle\product\10.1.0\admin\OBAY\cdump'
*.db_block_size=8192
*.db_cache_size=25165824
*.db_domain=''
*.db_file_multiblock_read_count=16
*.db_name='OBAY'
*.db_recovery_file_dest='D:\oracle\product\10.1.0\flash_recovery_area'
*.db_recovery_file_dest_size=2147483648
*.dispatchers='(PROTOCOL=TCP) (SERVICE=OBAYXDB)'
*.java_pool_size=50331648
*.job_queue_processes=10
*.large_pool_size=8388608
*.open_cursors=300
*.pga_aggregate_target=25165824
*.processes=150
*.remote_login_passwordfile='EXCLUSIVE'
*.shared_pool_size=83886080
*.sort_area_size=65536
*.undo_management='AUTO'
*.undo_tablespace='UNDOTBS1'
*.user_dump_dest='D:\oracle\product\10.1.0\admin\OBAY\udump'
 F1=Help                                              Line:3     Col:1
```

بعد التعديل نقوم بحفظ التعديلات على الملف .

3- عن طريق نظام التشغيل نقوم بعمل نسخ لاحد ملفات الControl Files الموجودة الان ونضعه ونسميه كما حددناه فى المتغير Control_Files .

```
Command Prompt

D:\oracle\product\101~1.0\oradata\OBAY>COPY CONTROL01.CTL CONTROL05.CTL
        1 file(s) copied.

D:\oracle\product\101~1.0\oradata\OBAY>
```

5- فتح قاعدة البيانات .

```
Command Prompt - SQLPLUS /NOLOG

SQL> STARTUP;
ORACLE instance started.

Total System Global Area  171966464 bytes
Fixed Size                   787988 bytes
Variable Size             145488364 bytes
Database Buffers           25165824 bytes
Redo Buffers                 524288 bytes
Database mounted.
Database opened.
SQL>
```

بالطبع يمكن التحقق من التعديل عن طريق الامر SHOW APRAMAETERS.

```
Command Prompt - SQLPLUS /NOLOG

SQL> SHOW PARAMETERS CONTROL_FILES;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
control_files                        string      D:\ORACLE\PRODUCT\10.1.0\ORADA
                                                 TA\OBAY\CONTROL01.CTL, D:\ORAC
                                                 LE\PRODUCT\10.1.0\ORADATA\OBAY
                                                 \CONTROL02.CTL, D:\ORACLE\PROD
                                                 UCT\10.1.0\ORADATA\OBAY\CONTRO
                                                 L03.CTL, D:\ORACLE\PRODUCT\10.
                                                 1.0\ORADATA\OBAY\CONTROL04.CTL
SQL>
```

تغيير إسم قاعدة البيانات :-

هل فكرت يوماً تغيير إسم قاعدة بياناتك ؛ لاشك أنك فكرت فى هذا لكن كيف الطريق إلى ذلك ، فى الحقيقة الامر جدُ يسير . فتغيير إسم قاعدة البيانات يلزم تعديل ملفين اثنين هما ملف المتغيرات Parameter File وملف التحكم Control File . واليك الخطوات :-

1- التأكد من إسم قاعدة البيانات الان عن طريق الاستعلام التالى :

```
SELECT NAME FROM V$DATABASE;
```

```
Command Prompt - sqlplus /nolog

SQL> SELECT NAME FROM V$DATABASE;

NAME
---------
OBAY

SQL>
```

إسم قاعدة البيانات الان OBAY.

2- الان نقوم نقوم بتغيير إسم قاعدة البيانات على ملف التحكم الControl File
. لكن المشكلة أن هذا الملف ليس نصياً حتى نستطيع تعديله ، فهو ملف ثنائى ،
لذا يمكن اللجوء لعمل نسخ احتياطى لهذا الملف كTrace File .
نتيجة هذا النسخ الاحتياطى Trace File عباره عن ملف نصى يحتوى على
كود يمكن ان يستعمل لإنشاء Control File جديد ، كما يمكن التعديل فى هذا
الكود . بحيث نغير اسم قاعدة البيانات . يوجد هذا الملف فى المسار المعرف فى
المتغير user_dump_dest .
والان نجرى عملية النسخ الإحتياطى .

```
ALTER DATABASE BACKUP CONTROLFILE TO TRACE;
```

```
Command Prompt - sqlplus /nolog

SQL> ALTER DATABASE BACKUP CONTROLFILE TO TRACE;

Database altered.

SQL>
```

عند فتح هذا الملف يكون شكله كالتالى :

```
obay_ora_3924 - Notepad
File  Edit  Format  View  Help
-- a script file, edited as necessary, and executed when there is a
-- need to re-create the control file.
--
--     Set #1. NORESETLOGS case
--
-- The following commands will create a new control file and use it
-- to open the database.
-- Data used by Recovery Manager will be lost.
-- Additional logs may be required for media recovery of offline
-- Use this only if the current versions of all online logs are
-- available.
-- After mounting the created controlfile, the following SQL
-- statement will place the database in the appropriate
-- protection mode:
--  ALTER DATABASE SET STANDBY DATABASE TO MAXIMIZE PERFORMANCE
STARTUP NOMOUNT
CREATE CONTROLFILE REUSE DATABASE "OBAY" NORESETLOGS  NOARCHIVELOG
    MAXLOGFILES 16
    MAXLOGMEMBERS 3
    MAXDATAFILES 100
    MAXINSTANCES 8
    MAXLOGHISTORY 454
LOGFILE
  GROUP 1 'D:\ORACLE\PRODUCT\10.1.0\ORADATA\OBAY\REDO01.LOG'  SIZE 10M,
  GROUP 2 'D:\ORACLE\PRODUCT\10.1.0\ORADATA\OBAY\REDO02.LOG'  SIZE 10M,
  GROUP 3 'D:\ORACLE\PRODUCT\10.1.0\ORADATA\OBAY\REDO03.LOG'  SIZE 10M
-- STANDBY LOGFILE
DATAFILE
  'D:\ORACLE\PRODUCT\10.1.0\ORADATA\OBAY\SYSTEM01.DBF',
  'D:\ORACLE\PRODUCT\10.1.0\ORADATA\OBAY\UNDOTBS01.DBF',
  'D:\ORACLE\PRODUCT\10.1.0\ORADATA\OBAY\SYSAUX01.DBF',
  'D:\ORACLE\PRODUCT\10.1.0\ORADATA\OBAY\USERS01.DBF'
CHARACTER SET WE8MSWIN1252
;
-- Commands to re-create incarnation table
-- Below log names MUST be changed to existing filenames on
-- disk. Any one log file from each branch can be used to
```

ما يهمنا فى هذا الملف هو ما قمنا بتظليله ، والان نقوم بعمل نسخ لهذا الجزء المظلل ونحفظه فى ملف جديد ولنفترض أن اسمه Control.trc .

بالطبع إذا قمنا بعمل نسخ احتياطى للControl File بالطريقة اعلاه ثم بعد ذلك قمنا بتغير تكوينات قاعدة البيانات مثلاً إضافة Tablespace أو Data File ، فيلزمنا تعديل الTrace File بالتعديلات الجديدة فى حال استخدمناه لإنشاء Control File.
يكون شكل الملف Control.trc كالاتى :

```
CREATE CONTROLFILE REUSE DATABASE "OBAY" NORESETLOGS
NOARCHIVELOG
    MAXLOGFILES 16
    MAXLOGMEMBERS 3
    MAXDATAFILES 100
    MAXINSTANCES 8
    MAXLOGHISTORY 454
LOGFILE
  GROUP 1
'D:\ORACLE\PRODUCT\10.1.0\ORADATA\OBAY\REDO01.LOG'  SIZE
10M,
  GROUP 2
'D:\ORACLE\PRODUCT\10.1.0\ORADATA\OBAY\REDO02.LOG'  SIZE
10M,
  GROUP 3
'D:\ORACLE\PRODUCT\10.1.0\ORADATA\OBAY\REDO03.LOG'  SIZE
10M
-- STANDBY LOGFILE
DATAFILE
  'D:\ORACLE\PRODUCT\10.1.0\ORADATA\OBAY\SYSTEM01.DBF',
  'D:\ORACLE\PRODUCT\10.1.0\ORADATA\OBAY\UNDOTBS01.DBF',
  'D:\ORACLE\PRODUCT\10.1.0\ORADATA\OBAY\SYSAUX01.DBF',
  'D:\ORACLE\PRODUCT\10.1.0\ORADATA\OBAY\USERS01.DBF'
CHARACTER SET WE8MSWIN1252
```

هذا هو الملف Control.trc الذى سنستخدمه لإنشاء الControl File الجديد ولكن قبل ذلك يلزمنا تعديل ما قمت بتظليله على النحو التالى :-

REUSE	→	**SET**
OBAY	→	**TEST (THIS NEW DATABASE NAME)**
NORESETLOGS	→	**RESETLOGS**

3- نقوم بإغلاق قاعدة البيانات .

```
Command Prompt - sqlplus /nolog

SQL> SHUT IMMEDIATE;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> _
```

4- تغيير إسم قاعدة البيانات على الملف INITobay.ora وهو ملف المتغيرات (PFILE) التى تعمل به قاعدة البيانات الحالية .

```
Command Prompt - EDIT INITOBAY.ORA
  File  Edit  Search  View  Options  Help
            D:\oracle\product\10.1.0\db_1\database\INITobay.ORA
*.background_dump_dest='D:\oracle\product\10.1.0\admin\OBAY\bdump'
*.compatible='10.1.0.2.0'
*.control_files='D:\oracle\product\10.1.0\oradata\OBAY\CONTROL01.CTL','D:\orac
*.core_dump_dest='D:\oracle\product\10.1.0\admin\OBAY\cdump'
*.db_block_size=8192
*.db_cache_size=25165824
*.db_domain=''
*.db_file_multiblock_read_count=16
*.db_name='OBAY'
*.db_recovery_file_dest='D:\oracle\product\10.1.0\flash_recovery_area'
*.db_recovery_file_dest_size=2147483648
*.dispatchers='(PROTOCOL=TCP) (SERVICE=OBAYXDB)'
*.java_pool_size=50331648
*.job_queue_processes=10
*.large_pool_size=8388608
*.open_cursors=300
*.pga_aggregate_target=25165824
*.processes=150
*.remote_login_passwordfile='EXCLUSIVE'
*.shared_pool_size=83886080
*.sort_area_size=65536
*.undo_management='AUTO'
 F1=Help                                          Line:9      Col:16
```

نغير إسم قاعدة البيانات db_name من OBAY الى TEST ثم نحفظ التعديلات .

5- مسح ملفات الControl Files الحالية فى قاعدة البيانات.

```
Command Prompt

D:\oracle\product\10.1.0\oradata\OBAY>DEL *.CTL
```

6- تشغيل قاعدة البيانات فى الوضع NOMOUNT .

```
SQL> STARTUP NOMOUNT;
ORACLE instance started.

Total System Global Area  171966464 bytes
Fixed Size                   787988 bytes
Variable Size             145488364 bytes
Database Buffers           25165824 bytes
Redo Buffers                 524288 bytes
SQL>
```

7- الان نقوم بإنشاء الControl File باستخدام الTrace File الذى قمنا بانشاءه سابقاً .

```
SQL> @C:\CONTROL.TRC;

Control file created.

SQL>
```

8- نفتح قاعدة البيانات فى الوضع RESETLOGS.

```
SQL> ALTER DATABASE OPEN RESETLOGS;

Database altered.

SQL>
```

9- للتأكد من تغيير اسم قاعدة البيانات

```
SQL> SELECT NAME FROM V$DATABASE;

NAME
---------
TEST

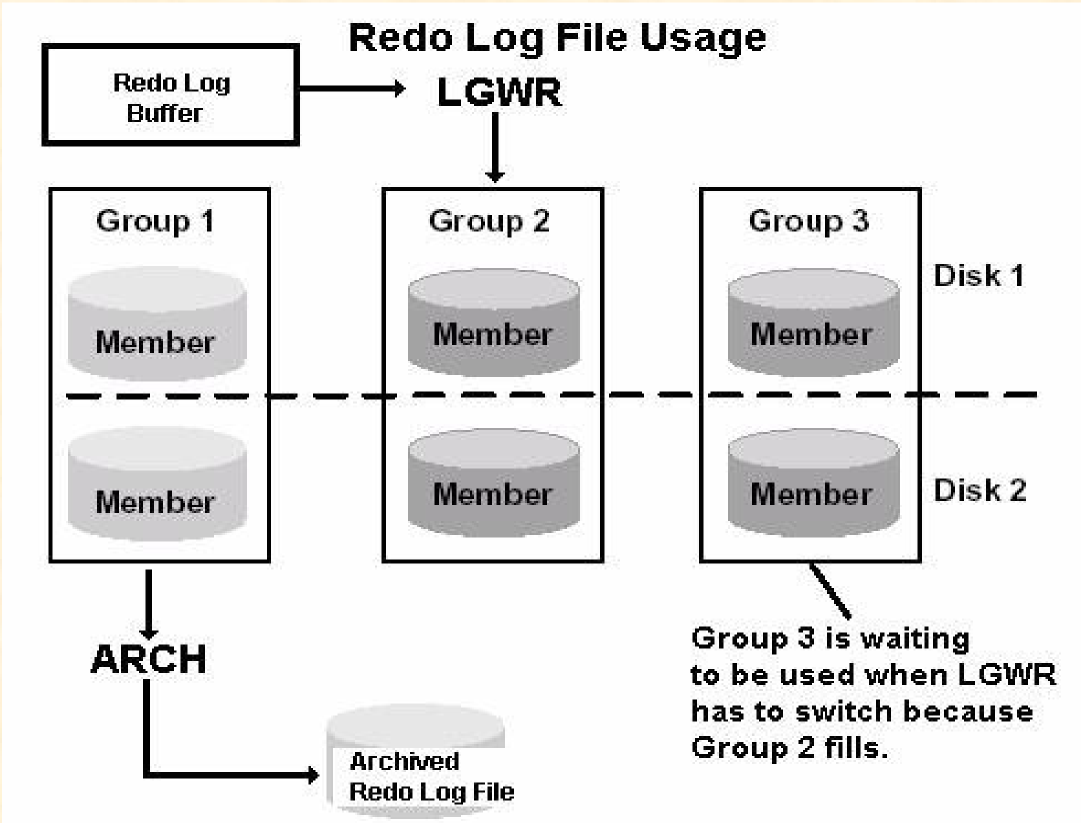
SQL>
```

Quick Reference

Context	Reference
Initialization parameters	CONTROL_FILES
Dynamic performance views	V$CONTROLFILE V$CONTROLFILE_RECORD_SECTION V$PARAMETER
Data dictionary views	None
Commands	None
Packaged procedures and functions	None

:Redo Log Files

هذ



التغييرات التى تم تثبيتها او لا ، نستفيد من هذا الملفات فى استرجاع قاعدة البيانات فى حال حدوث فشل . بحيث تكون هناك بعض البيانات لم تكتب فى ال Data Files بينما كتبت فى هذا الملفات .

يكون هذا الملف فى شكل GROUP بحيث تعمل كل قاعدة البيانات على على الاقل باثنين من الGroups .

إذاً يستخدم هذا الملف اساساً لعملية الاسترجاع فى حالة حدوث خطأً إذ يحتوى على التغييرات التى تحصل على البيانات .

ولضمان الحفاظ على هذا الملف فإن كل Group ينتظم فى شكل Members وكل الMember داخل الGroup هى نسخة طبق الاصل الهدف منها تقليل نسبة خطر الفقدان .

LGWR Background Process يقوم بكتابة البيانات الموجود الRedo log Buffer الى الRedo Log Files.

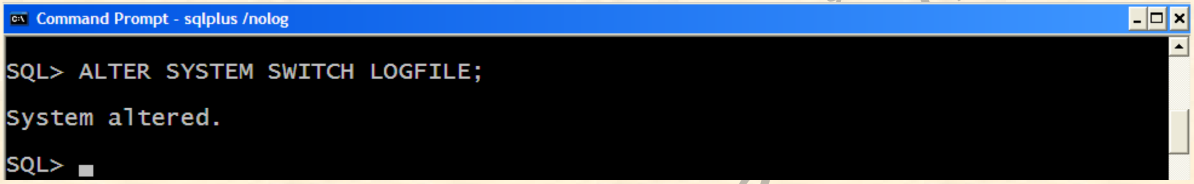
لحظة ملئ الRedo Log File فان الLGWR يقوم بالتحول للRedo Log File الاخرى فى عملية تعرف بالLog Switch .

إذاً فالOracle Server يقوم بكتابة كل التغيرات على قاعدة البيانات فى ال Redo Log Buffer . هذه المتغيرات تكتب من الBuffer إلى الRedo Log File عن طريق الLGWR فى الحالات التالية :-

1- لحظة تثبيت العمليات (Commit).
2- لحظة إمتلاء الBuffer الى الثلث .
3- قبل عملية DBWn.

إذاً عملية الLog Switch تحدث إلياً دون تدخل من مدير قاعدة البيانات ، لكن قد يحتاج مدير قاعدة البيانات احياناً لعمل هذه المهمة يدوياً من خلال الامر التالى -:

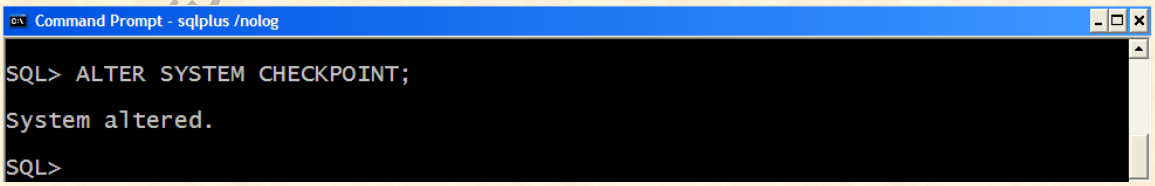
```
ALTER SYSTEM SWITCH LOGFILE;
```



كذلك عملية الCheckpoint تحدث الياً عند حدوث الLog Switch وهى تقوم بعمل تعديل للHeader لكل الData Files والControl Files للإشارة بأن اكمل بنجاح .

ايضاً يمكن عمل Checkpoint الياً عن طريق مدير قاعدة البيانات بالامر التالى -:

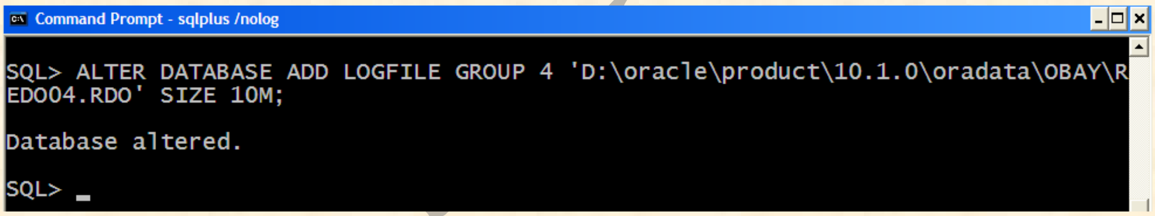
```
ALTER SYSTEM CHECKPOINT;
```



إضافة Online Redo Log Files Group :

فى بعض الاحيان نحتاج لإضافة Online Redo Log Files Group جديد ، وذلك عن طريق الامر التالى :-

```
ALTER DATABASE ADD LOGFILE GROUP 4 ‘D:\ORACLE\
PRODUCT\10.1.0\ORADATA\OBAY\REDO04.RDO’ SIZE
10M;
```



ولإضافة Members للGroups :-

```
ALTER DATABASE ADD LOGFILE MEMBER 'D:\oracle\
product\10.1.0\oradata\OBAY\RED1.RDO' TO GROUP
1, ,'D:\oracle\product\10.1.0\oradata\OBAY\
RED2.RDO' TO GROUP 2, 'D:\oracle\product\10.1.0\
oradata\OBAY\RED3.RDO' TO GROUP  3,'D:\oracle\
product\10.1.0\oradata\OBAY\RED4.RDO' TO GROUP
4;
```

```
SQL> ALTER DATABASE ADD LOGFILE MEMBER 'D:\oracle\product\10.1.0\oradata\OBAY\RE
D1.RDO' TO GROUP 1 ,'D:\oracle\product\10.1.0\oradata\OBAY\RED2.RDO' TO GROUP 2,
'D:\oracle\product\10.1.0\oradata\OBAY\RED3.RDO' TO GROUP 3,'D:\oracle\product\1
0.1.0\oradata\OBAY\RED4.RDO' TO GROUP 4;

Database altered.

SQL>
```

حذف Online Redo Log Groups:-

```
ALTER DATABASE DROP LOGFILE GROUP 4;
```

مع ملاحظة أن قاعدة البيانات تعمل على الاقل على اثنين من الGroups ،

```
SQL> ALTER DATABASE DROP LOGFILE GROUP 4;

Database altered.

SQL>
```

كذلك لا يمكن حذف current or active Groups .

أيضاً عند حذف الGroup عن طريق الOracle تظل الملفات موجودة على مستوى نظام التشغيل .

حذف Online Redo Log Members:-

```
ALTER DATABASE DROP LOGFILE MEMBER ‘D:\ORACLE\
PRODUCT\10.1.0\ORADATA\OBAY\RED3.RDO;
```

```
SQL> ALTER DATABASE DROP LOGFILE MEMBER 'D:\oracle\product\10.1.0\oradata\OBAY\R
ED3.RDO';

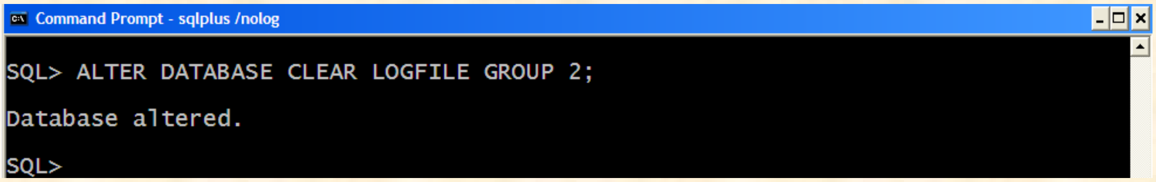
Database altered.

SQL>
```

Clearing Online Redo Log Files:-

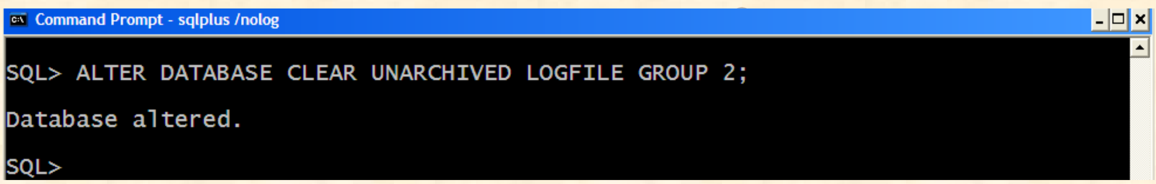
وهى لحل مشكلة العطل الذى يحدث لكل الMembers فى الGroup ، فهى ضمناً تعنى إعادة إنشاء للRedo Log Files .

```
ALTER DATABASE CLEAR LOGFILE GROUP 2;
```



إذا كانت قاعدة البيانات تعمل فى الوضع Archive log فإن عملية الClearing اعلاه ستفشل إذا كان الGroup 2 لم تتم ارشفته ، عموماً سنتحدث عن موضوع الارشيف لاحقاً ولكن لحل هذا الإشكال يمكن تعديل الامر السابق للاتى :-

```
ALTER DATABASE CLEAR UNARCHIVED LOGFILE GROUP 2;
```



ولكن يجب قبل القيام بهذا الامر بعمل نسخ إحتياطى كامل لقاعدة البيانات (Full Backup) .

إعادة تسمية الRedo Log Files:-

يمكن إعادة تسمية الRedo Log Files او تغيير مكانه عن طريق إضافة Log File جديد وحذف القديم . يمكن كذلك إستخدام الامر ALTER DATABASE RENAME FILE لكن هذا الامر يتطلب جلب قاعدة البيانات فى الوضع MOUNT.

فقدان الRedo Log File:

إذا فقدنا Member من الRedo Log Files Group فإن قاعدة البيانات ستظل مفتوحة ومتاحة ما دام أن هناك Member اخرى فى الGroup ،

لكن سوف نستقبل رسائل تحذيرية فى الAlert log File . بالطبع يمكن الإسترجاع بعمل نسخ لاحد الMember الموجود ومن ثم اعادة تسميتها .

للإستلام عن الRedo Log File:

- V$LOG
- V$LOGFILE

الفصل السادس
إدارة المستخدمين
Administering Users

سنتطرق قى هذا الفصل لكل من الاتى:-

- **إنشاء وإدارة المستخدمين فى قاعدة البيانات اوركل .**
- **إنشاء وإدارة الوظائف Roles .**
- **منح وسحب الصلاحيات Privileges.**

- **التحكم فى الموارد Resource المستخدمة عن طريق المستخدمين .**

وهذا الفصل من الاهمية بمكان ، بحيث يعتبر من اهم اولويات مدير قاعدة البيانات .

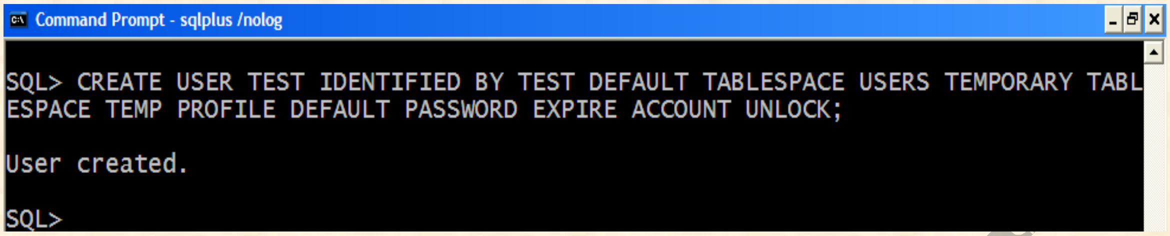
فالفكرة الاساسية تقوم على أن الوصول لقاعدة البيانات يتطلب وجود مستخدم يمتلك صلاحية الوصول ، إذا لا يمكن ان تظل قاعدة البيانات متاحة للجميع .

واول ما يمكن مناقشته هنا كيفية إنشاء مستخدم فى قاعدة البيانات . فمدير قاعدة البيانات يجب مراعاة الاتى عند إنشاء المستخدمين :-
1- ان يكون اسم المستخدم فريداً فى قاعدة البيانات وأن لا يتعدى عدد حروفه 30 حرفاً ، وايضاً لا يحتوى على مساحات ويبتدئ بحرف اى ليس رقما مثلاً .
2- وسيلة التحقق . سواء اكان ذلك عن طريق قاعدة البيانات او عن طريق نظام التشغيل او غيره .
3- Default Tablespace وهو الTablespace الذى يستخدم لإنشاء وتخزين كائنات المستخدم .
4- Temporary Tablespace وهو الTablespace الذي يستخدم لإنشاء الكائنات المؤقتة للمستخدم .
5- الProfile للمستخدم وهو لإدارة الموارد وكلمة السر للمستخدم .

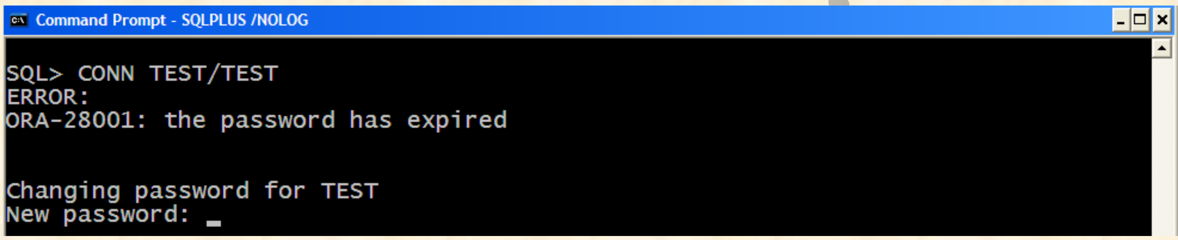
عند إنشاء مستخدم فى قاعدة البيانات يتم خلق Schema وهى مجموعة الكائنات الممتلكة للمستخدم وتكون اسم الSchema بنفس اسم المستخدم .
ولنفترض اننا الان نريد إنشاء مستخدم TEST .

```
CREATE USER TEST IDENTIFIED BY TEST DEFAULT
TABLESPACE USERS TEMPORARY TABLESPACE TEMP
PROFILE DEFAULT PASSWORD EXPIRE ACCOUNT UNLOCK;
```

فى السناريو السابق قمنا بخلق مستخدم اسمه TEST وكلمة السر لهذا المستخدم ايضاً TEST ، هذه المستخدم ينشئ ويخزن الكائنات فى Users



Tablespace ، كما يخزن الكائنات المؤقتة فى Temp Tablespace . هذا المستخدم يجب عليه تغيير كلمة السر عند اول دخول كما نشاهد ادناه ، ايضا هذا المستخدم ACCOUNT غير مغلق اى يمكن الاتصال بعد تغيير كلمة السر.



بالطبع قد لانحتاج لتحديد الDefault Tablespace وكذلك الTemporary Tablespace.

بمعنى اخر إذا كان Default Temporary Tablespace فى قاعدة البيانات هو Temp وكان الDefault Tablespace فى قاعدة البيانات هو Users فلا يلزمنا تحديد ذلك عند إنشاء المستخدم إذا كنا لا نريد تغيير الDefault.

كذلك الحال بالنسبة للProfile فإنه إذا لم يتم تحديده فسوف يُمنح الDefault Profile .

:QUOTA

وهى الحصة المتاحة للمستخدم لإنشاء الكائنات فى الTablespaces وبدونها يكون المستخدم عاجز عن إنشاء اى كائن .

لذا يمكن لمدير قاعدة البيانات منع المستخدم من إنشاء أى كائن عن طريق منحه حصة Quota تساوى صفر فى الTablespace .

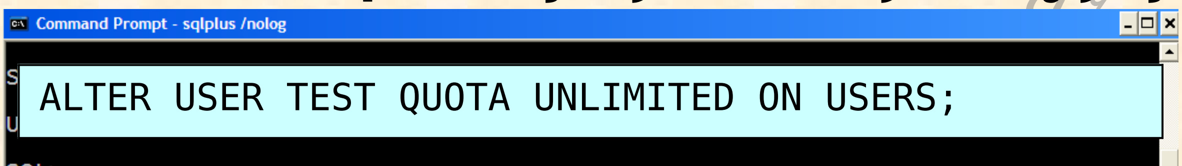
كذلك يستطيع مدير قاعدة البيانات منح المستخدم حصة Quota غير محدودة فى الTablespace عن طريق منحه Unlimited .

يمكن منح الحصة للمستخدم اثناء الإنشاء او بعد ذلك .

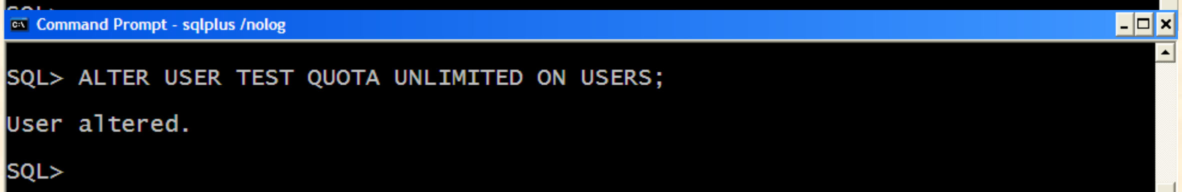
فلنفترض الان اننا نريد منح حصة تساوى صفر للمستخدم TEST فى Users Tablespace .

```
ALTER USER TEST QUOTA 0M ON USERS;
```

ولنفترض الان اننا نريد منحه حصه غير محدودة فى Users Tablespace .

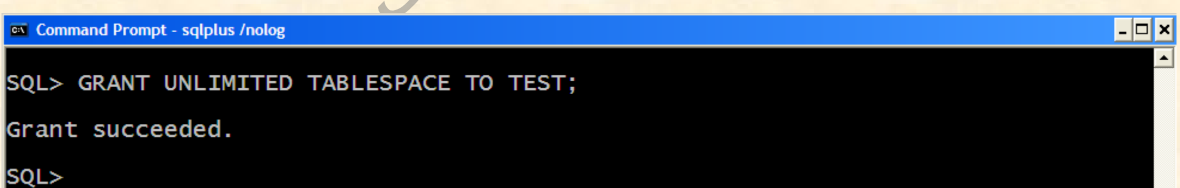



```
ALTER USER TEST QUOTA UNLIMITED ON USERS;
```



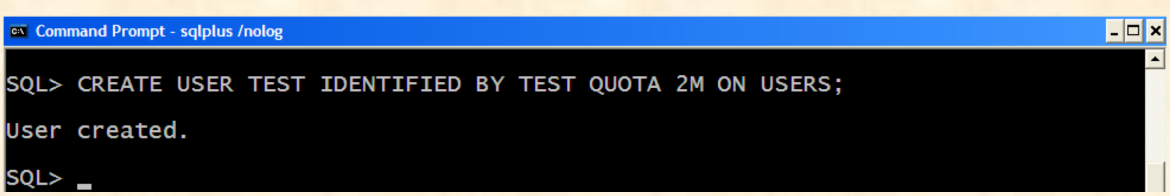
هناك خيار اخر وهو منح هذا المستخدم صلاحية تسمى Unlimited Tablespace . وهى صلاحية تسمح لهذا المستخدم حصة غير محدودة فى الTablespaces.

```
ALTER USER TEST QUOTA UNLIMITED ON USERS;
```



كما ذكرنا سابقاً يمكن منح الحصة اثناء خلق إنشاء المستخدم كالاتى :-

```
CREATE USER TEST IDENTIFIED BY TEST QUOTA 2M ON
USERS;
```



يمكن الاستعلام عن الحصة عن طريق :-
DBA_TS_QUOTAS
USER_TS_QUOTAS

وسيلة التحقيق :-

وهى طريقة التحقق من المستخدمين للوصول لقاعدة البيانات ، فلحظة إنشاء المستخدم لابد من اختيار وسيلة التحقق :-

كلمة السر (Password) : وهى الطريقة المستخدمة فى السناريوهات السابقة ، بحيث يكون التحقق عن طريق قاعدة البيانات اوركل عند محاولة الدخول فيتم التحقق من كلمة السر .
وهى تكون كالاتى :- identified by password.

External: ويكون التحقق عن طريق نظام التشغيل بحيث يستطيع المستخدم الاتصال بقاعدة البيانات دون إدخال اسم المستخدم وكلمة السر لقاعدة البيانات. وذلك لأن كلمة السر لقاعدة البيانات غير مستخدمة فى هذا النوع من التحقق .
يتطلب هذا النوع من التحقق تهيئة المتغير OS_AUTHENT_PREFIX بحيث يحتوى على قيمة حرفية هى OPS$ فى الاصل ، بالطبع نستطيع تغيره كما نشاء ، هذا المتغير نحتاجه عند عملية التحقق من المستخدم الذى يريد الإتصال بقاعدة البيانات .
ولنفترض أننا قمنا بتغيير المتغير اعلاه الى القيمة OBAY ، ومستخدم نظام التشغيل الذى نستخدمه هو ADMINISTRATOR إذاً يلزمنا إنشاء مستخدم فى قاعدة البيانات اسمه OBAYADMINISTRATOR ونختار طريق التحقق External .

تابع معى الخطوات :-
1- نتأكد من قيمة المتغير OS_AUTHENT_PREFIX

```
SHOW PARAMETER OS_AUTHENT_PREFIX;
```

```
Command Prompt - sqlplus /nolog
SQL> SHOW PARAMETER OS_AUTHENT_PREFIX;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------
os_authent_prefix                    string      OPS$
```

2- نقوم بتغير المتغير الى القيمة OBAY مثلاً

```
ALTER SYSTEM SET OS_AUTHENT_PREFIX='OBAY'
SCOPE=SPFILE;
```

```
Command Prompt - sqlplus /nolog

SQL> ALTER SYSTEM SET OS_AUTHENT_PREFIX='OBAY' SCOPE=SPFILE;

System altered.

SQL>
```

بالطبع نحتاج لإغلاق قاعدة البيانات وفتحها من جديد .

3- الان يمكن التأكد من قيمة التغير .

```
Command Prompt - sqlplus /nolog
SQL> SHOW PARAMETER OS_AUTHENT_PREFIX;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
os_authent_prefix                    string      OBAY
SQL>
```

4- الان نقوم بخلق المستخدم OBAYADMINISTRATOR فى قاعدة البيانات .

```
CREATE USER OBAYADMINISTRATOR IDENTIFIED
EXTERNALLY;
```

```
Command Prompt - sqlplus /nolog

SQL> CREATE USER OBAYADMINISTRATOR IDENTIFIED EXTERNALLY;

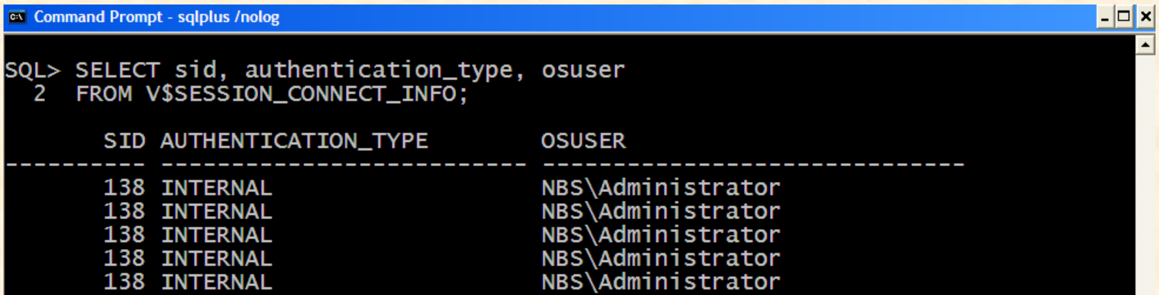
User created.

SQL>
```

يمكن الإستعلام عن المستخدمين فى قاعدة البيانات عن طريق :
DBA_USERS

كما يمكن معرفة نوع التحقق عن الSESSION المفتوحة فى قاعدة البيانات
الان المستخدم OBAYADMINISTRATOR يستطيع الإتصال بقاعدة البيانات

```
SELECT sid,authentication_type, osuser FROM
V$SESSION_CONNECT_INFO;
```



مباشرة ودون التحقق من قاعدة البيانات فالتحقق يتم عن طريق نظام التشغيل.

الصلاحيات (Privilege):

كل المستخدمين الذين قمنا بخلقهم فى السناريوهات السابقة لا يستطيعون الاتصال بقاعدة البيانات فضلاً عن القيام بأى مهام اخرى إذ ليس لديهم صلاحيات لذلك .

فى الخطوات السابقة قمنا بخلق المستخدم وحددنا له طريقة التحقق ومنحناه الحصة المطلوبة وحددنا له الProfile لادارة كلمة السر والموارد ، لكن ينتظرنا الان أن نعطيه صلاحيات للعمل فى قاعدة البيانات . ويكون منح الصلاحيات عن طريق مدير قاعدة البيانات او ممن يملك الصلاحيات .

هناك نوعن من الصلاحيات :-

1- System:- وهو لتمكين المستخدمين من انجاز اعمال معينة على قاعدة البيانات .

وهناك اكثر من 700 صلاحية System Privileges على قاعدة البيانات ، على سبيل المثال :

SYSDBA & SYSOPER : هذه الصلاحيات تسمح للمستخدم إغلاق وفتح قاعدة البيانات.

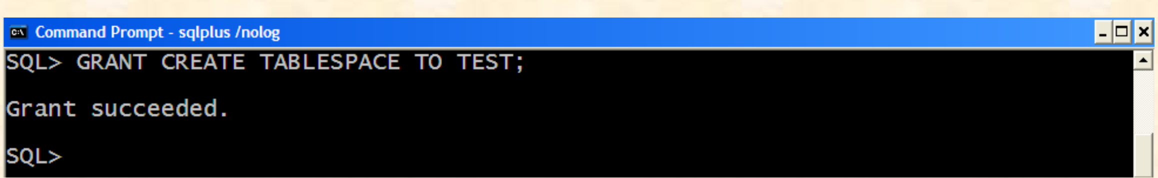
CREATE TABLESPACE : تسمح للمستخدم إنشاء Tablespace.

عموماً يمكن استعراض كل الصلاحيات System Privileges المتاحة لمدير قاعدة البيانات عن طريق الاستعلام الاتى :-

```
SELECT PRIVILEGE FROM DBA_SYS_PRIVS;
```

ولنفترض أننا نريد منح بعض الصلاحيات System Privileges للمستخدم TEST .

```
GRANT CREATE TABLESPACE TO TEST;
```



هكذا منحنا المستخدم TEST صلاحية إنشاء الTablespace .

يجب مراعاة الكلمة ANY عند منح الصلاحيات ، ولتوضيح هذا المعنى نلاحظ الفرق بين الصلاحيتين :
SELECT TABLE: تسمح للمستخدم عمل SELECT على الجداول التى يملكها المستخدم.
SELECT ANY TABLE: تسمح للمستخدم عمل استعلام SELECT على الجداول التى يملكها المستخدمين الاخرين.

هناك خيار عند منح الصلاحيات System Privileges للمستخدم وهو الخيار WITH ADMIN OPTION.
هذا الخيار يعنى ان المستخدم بعد أن يملك الصلاحية يستطيع منحها لغيره من المستخدمين .

```
GRANT ALTER SYSTEM TO TEST WITH ADMIN OPTION;
```

```
Command Prompt - sqlplus /nolog
SQL> GRANT ALTER SYSTEM TO TEST WITH ADMIN OPTION;

Grant succeeded.

SQL>
```

المستخدم TEST الان يملك الصلاحية ALTER SYSTEM ويستطيع منحها لغيره من المستخدمين .
هناك بعض الصلاحيات التى يجب ان لا تمنح الى لمديرى قاعدة البيانات ويجب مراعاة السرية عند المنح مثلاً (SYSDBA & SYSOPER & RESTRICTED SESSION & ALTER DATABASE) وغيره من الصلاحيات التى يجب أن تكون فقط لمدراء قاعدة البيانات ..
بالطبع يمكن سحب الصلاحيات من المستخدمين ، ولنفترض أننا نريد سحب صلاحية CREATE TABLESPACE من المستخدم TEST.

```
REVOKE CREATE TABLESPACE FROM TEST;
```

تخيل معى هذا السناريو وهو أن مدير قاعدة البيانات منح المستخدم TEST

```
Command Prompt - sqlplus /nolog
SQL> REVOKE CREATE TABLESPACE FROM TEST;

Revoke succeeded.

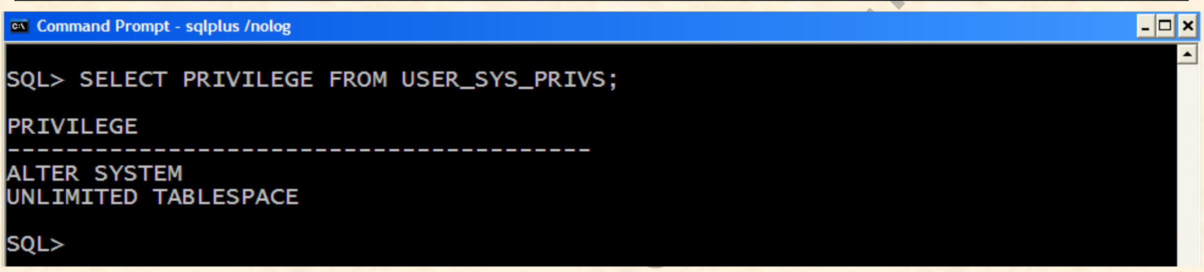
SQL> _
```

صلاحية CREATE TABLESPACE عن طريق الخيار WITH ADMIN

OPTION ، قام المستخدم TEST بإنشاء Tablespace ومن ثم منح هذه الصلاحية لمستخدم اخر اسمه TEST1.
اراد مدير قاعدة البيانات سحب الصلاحية CREATE TABLESPACE من المستخدم TEST.
اولا سيستم سحب الصلاحية من المستخدم دون التاثير على المهام التى نفذها بواسطة هذه الصلاحية وهى هنا انشاء الTablespace.
ثانياً : المستخدم TEST1 لا يتاثر بسحب الصلاحيات من المستخدم TEST

هذا المستخدم يستطيع معرفة ما يملكه من صلاحيات System Privileges عن طريق الإستعلام :-

```
SELECT PRIVILEGE FROM USER_SYS_PRIVS;
```



للإستعلام :

DBA_ROLES
USER_ROLE_PRIVS
DBA_ROLE_PRIVS
ROLE_ROLE_PRIVS
ROLE_SYS_PRIVS
ROLE_TAB_PRIVS
SESSION_PRIVS

2- Object Privileges:- وهو لتمكين المستخدم للوصول واستعمال الكائن المعين ، بدون هذه الصلاحيات المستخدم له الصلاحيات على الكائنات التى يملكها فقط .

المستخدم فى قاعدة البيانات له صلاحيات بالطبع على الكائنات التى يملكها ، يستطيع المستخدم ان يمنح صلاحيات لمستخدم اخر للوصول للكائنات التى يملكها ، وكذلك مدير قاعدة البينات يستطيع منح صلاحيات للمستخدمين للوصول لكائنات يملكها مستخدم اخر .

نفترض ان المستخدم TEST يملك جدول اسمه EXAMPLE ، هذا المستخدم يريد منح صلاحية SELECT على هذا الجدول للمستخدم TEST1.

```
GRANT SELECT ON EXAMPLE TO TEST1;
```

ماذا لو اراد مدير قاعدة البيانات منح صلاحية SELECT على الجدول

```
Command Prompt - sqlplus /nolog
SQL> GRANT SELECT ON EXAMPLE TO TEST1;

Grant succeeded.

SQL> _
```

EXAMPLE الذى يمتلكه المستخدم TEST للمستخدم TEST1.

```
GRANT SELECT ON TEST.EXAMPLE TO TEST1;
```

```
Command Prompt - sqlplus /nolog
SQL> GRANT SELECT ON TEST.EXAMPLE TO TEST1;

Grant succeeded.

SQL> _
```

كذلك يمكن استخدام الخيار WITH GRANT OPTION عند منح الصلاحيات Objects Privileges إشارة إلى أن هذا المستخدم بعد ان يملك هذه الصلاحية يمكن أن يمنحها غيره من المستخدمين .

```
GRANT INSERT ON EXAMPLE TO TEST1 WITH GRANT
OPTION;
```

```
Command Prompt - sqlplus /nolog
SQL> GRANT INSERT ON EXAMPLE TO TEST1 WITH GRANT OPTION;

Grant succeeded.

SQL> _
```

الان المستخدم TEST1 يستطيع منح صلاحية SELECT على الجدول EXAMPLE الذى يملكه المستخدم TEST لغيره من المستخدمين .

ماذا لو اراد المستخدم TEST سحب صلاحية الSELECT على الجدول EXAMPLE الذى يملكه من المستخدم TEST1 مع ملاحظة ان المستخدم TEST1 منح هذه الصلاحية لغيره من المستخدمين ولنفترض انه TEST2 ، النتيجة هى ان الصلاحية ستسحب من المستخدم TEST1 وكذلك كل المستخدمين الممنوحين الصلاحية عن طريق المستخدم TEST1 ؛ وهنا هو المستخدم TEST2 .

سنتابع هذا السناريو عملياً :-

```
SHOW USER;
```

```
SQL> SHOW USER;
USER is "TEST"
SQL> _
```

نحن الان نعمل على المستخدم TEST.

```
SQL> SELECT * FROM TAB;

TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
EXAMPLE                        TABLE

SQL>
```

المستخدم TEST يملك جدولاً واحداً هو EXAMPLE.
الان المستخدم TEST سيمنح صلاحية SELECT على الجدول EXAMPLE الذى يمتلكه للمستخدم TEST1.

```
GRANT SELECT ON EXAMPLE TO TEST1 WITH GRANT OPTION;
```

```
SQL> GRANT SELECT ON EXAMPLE TO TEST1 WITH GRANT OPTION;

Grant succeeded.

SQL>
```

الان عن طريق المستخدم TEST1 يمكن عمل SELECT على الجدول اعلاه :

```
Command Prompt - sqlplus /nolog
SQL> SHOW USER
USER is "TEST1"
SQL>
```

```
Command Prompt - sqlplus /nolog
SQL> SELECT * FROM TEST.EXAMPLE;

       CNO
----------
         1

SQL>
```

الان المستخدم TEST1 يستطيع منح صلاحية SELECT على الجدول EXAMPLE الذى يمتلكه المستخدم TEST للمستخدم TEST2.

```
GRANT SELECT ON TEST.EXAMPLE TO TEST2;
```

```
Command Prompt - sqlplus /nolog
SQL> GRANT SELECT ON TEST.EXAMPLE TO TEST2;

Grant succeeded.

SQL>
```

```
SELECT * FROM TEST.EXAMPLE;
```

الان المستخدم TEST2 يستطيع عمل SELECT على الجدول EXAMPLE .

```
Command Prompt - sqlplus /nolog
SQL> SHOW USER
USER is "TEST2"
SQL>
```

الان نقوم بسحب الصلاحية من المستخدم TEST1 .

```
REVOKE SELECT ON EXAMPLE FROM TEST1;
```

```
SQL> SELECT * FROM TEST.EXAMPLE;
```

```
Command Prompt - sqlplus /nolog
SQL> REVOKE SELECT ON EXAMPLE FROM TEST1;

Revoke succeeded.

SQL>
```

عن طريق المستخدم TEST1 نقوم بعمل SELECT للجدول EXAMPLE .

```
Command Prompt - sqlplus /nolog

SQL> SHOW USER
USER is "TEST1"
SQL>
```

لاحظ ان المستخدم فقد صلاحية عمل استعلام SELECT على الجدول EXAMPLE .

```
SELECT * FROM TEST.EXAMPLE;
```

```
Command Prompt - sqlplus /nolog

SQL> SELECT * FROM TEST.EXAMPLE;
SELECT * FROM TEST.EXAMPLE
                   *
ERROR at line 1:
ORA-01031: insufficient privileges


SQL>
```

ماذا عن المستخدم TEST2.

```
Command Prompt - sqlplus /nolog

SQL> SHOW USER
USER is "TEST2"
SQL>
```

```
SELECT * FROM TEST.EXAMPLE;
```

```
Command Prompt - sqlplus /nolog

SQL> SELECT * FROM TEST.EXAMPLE;
SELECT * FROM TEST.EXAMPLE
                   *
ERROR at line 1:
ORA-00942: table or view does not exist


SQL>
```

كذلك المستخدم TEST2 لا يملك صلاحية الإستعلام على الجدول EXAMPLE .
يمكن منح الصلاحية على مستوى العمود ، مثلاً :

```
Command Prompt - sqlplus /nolog

SQL> GRANT INSERT (CNO) ON EXAMPLE TO TEST1;

Grant succeeded.

SQL>
```

يمكن الاستعلام عن الObjects Privileges بعدة طرق منها :

DBA_TAB_PRIVS
ALL_TAB_PRIVS
USER_TAB_PRIV
DBA_COL_PRIVS
ALL_COL_PRIVS
USER_COL_PRIVS
SESSION_PRIVS

-:ROLES

تخيل معى انك تعمل مدير لقاعدة بيانات بها اكثر من 100 مستخدم ، هؤلاء المستخدمين على مستويات مختلفة من الصلاحيات ، ولنفترض انهم على خمسة مستويات .

ما العمل إذاً ؟ هل تمنح كل مستخدم صلاحياته بمفرده ؟ يلزمك إذاً عملاً شاقاً طويلاً .
الحل بكل بساطة هو فى الROLES.

الفكرة هى انه يمكن دمج مجموعة من الصلاحيات فى كائن واحد يسمى ROLE ، فالسناريو السابق لمدير قاعدة البينات الذى يدير قاعدة بيانات بها اكثر من 100 مستخدم على مستويات مختلفة وهى خمسة مستويات ابسط مما تتخيل ، يلزمنا الامرإنشاء خمسة من الROLES بعد ذلك نمنح الصلاحيات لهذه ال ROLES حسب المستويات ومن ثم منحها للمستخدمين .
هذا الحل يسهل علينا عملية ادارة الصلاحيات فبدل من ادارة اكثر من 100 مستخدم يلزمنا الامر ادارة خمسة من الROLES.
كذلك عند تعديل اى من الROLES فإن التعديل ينعكس اليا على المستخدمين فيسهل علينا عملية التعديل ، كذلك هذه العملية تحسن الأداة .

ولنفترض اننا نريد إنشاء ROLE جديدة تسمى OBAY، هذه الROLE نمنحها بعض الصلاحيات ولتكن CONNECT & RESOURCE ، بعد ذلك نمنح هذه الROLE للمستخدم TEST.

```
CREATE ROLE OBAY;
```

```
Command Prompt - sqlplus /nolog
SQL> CREATE ROLE OBAY;

Role created.

SQL>
```

يجب مراعاة أن إسم الROLE ان يكون فريداً فى قاعدة البيانات بحيث يكون الاسم لم يستعمل كمستخدم ولا كRoles فى قاعدة البيانات.

بالطبع يمكن انشاء ROLE ومنحها درجة من السرية عن طريق منحها كلمة سر للتحقق .

```
CREATE ROLE ROLENAME IDENTIFIED BY PASSWORD;
```

الان نمنح هذه الROLE بعض الصلاحيات .

```
GRANT CONNECT,RESOURCE TO OBAY;
```

```
Command Prompt - sqlplus /nolog
SQL> GRANT CONNECT,RESOURCE TO OBAY;

Grant succeeded.

SQL>
```

نمنح هذه الROLE للمستخدم TEST.

```
GRANT OBAY TO TEST;
```

```
SQL> GRANT OBAY TO TEST;
Grant succeeded.
SQL>
```

ملاحظة : يمكن اختيار الخيار WITH ADMIN OPTION عند منح الROLE .
هذا الخيار يمكن المستخدم بعد امتلاك الROLES منحها لغيره من المستخدمين .

يمكن سحب الROLE من المستخدم بنفس طريقة سحب الSystem Privilege .

```
REVOKE OBAY FROM TEST;
```

```
SQL> REVOKE OBAY FROM TEST;
Revoke succeeded.
SQL>
```

كذلك يمكن اجراء تعديلات على الROLES ولنفترض مثلاً اننا نريد عمل كلمة سر لل OBAY ROLE .

```
ALTER ROLE OBAY IDENTIFIED BY OBAY;
```

```
SQL> ALTER ROLE OBAY IDENTIFIED BY OBAY;
Role altered.
SQL>
```

```
SQL> DROP ROLE OBAY;
Role dropped.
SQL>
```

يمكن حذف الROLE عن طريق الامر .

```
DROP ROLE OBAY;
```

عند حذف الROLE يقوم ORACLE SERVER بحذف هذه الROLE من كل المستخدمين الذين مُنحوا هذا الROLE الياً.

:PROFILES

وهو لإدارة كلمة السر وكذلك الموارد ، فبعد كم سيتم انهاء كلمة السر؟ وماهى عدد المحاولات الفاشلة قبل ان يتم اغلاق حسابك ؟ وكم SESSION يسمح للمستخدم فتحها فى قاعدة البيانات؟ وغيره من الاسئلة التى يجب عنها ال PROFILE.

وكما ذكرنا سابقاً اى مستخدم فى قاعدة البيانات لابد أن يكون له PROFILE إذا لم يتم تحديد ذلك عند الإنشاء فان الORCLE SERVER سوف يمنح المستخدم PROFILE يسمى DEFAULT هذا الPROFILE يتم خلقه مع إنشاء قاعدة البيانات ولا يمن حزفه من قاعدة البيانات .

بالطبع يمكن إنشاء وتعديل وحذف PROFILE من قاعدة البيانات .

والسؤال الان ماهى النقاط التى يديرها الPROFILE بالنسبة لكلمة السر ؟

:FAILED_LOGIN_ATTEMPTS وهى عدد المحاولات الفاشلة للمستخدم لمحاولة دخول قاعدة البيانات قبل اغلاق حسابه.

:PASSWORD_LIFE_TIME لتحديد عدد الايام التى يستطيع بها المستخدم دخول قاعدة البيانات بكلمة السر قبل إنتهاءها.

:PASSWORD_REUSE_TIME لتحديد عدد الايام قبل استخدام كلمة السر مرة اخرى. اذا تم تحديد هذا الخيار يجب ان يكون الخيار PASSWORD_REUSE_MAX=UNLIMITED.

:PASSWORD_REUSE_MAX لتحديد عدد كلمات السر التى تم تغيرها قبل استخداها مرة اخرى. اذا اتم تحديد هذا الخيار يجب أن يكون الخيار PASSWORD_REUSE_TIME=UNLIMITED.

:PAWWORD_LOCK_TIME لتحديد عدد الايام قبل قفل هذا المستخدم.

:PASSWORD_GRACE_TIME لتحديد عدد الايام التى بعدها يتم اصدار تحذيرات أنه بعد هذه الفترة سيتم إنهاء كلمة السر الحالية.

:PASSWORD_VERFY_FUNTION لتحديد الدالة التى ترسم سياسة اختيار كلمة السر.

أما بالنسبة بالمصادر المتاحة فيتم تحديد الاتى :
CONNECT TIME: لتحديد الدقائق المسموح بها للمستخدم المتصل بقاعدة البيانات قبل قطع اتصاله الياً.
IDLE TIME: لتحديد الدقائق المسموح بها للمستخدم المتصل بقاعدة البيانات ان يظل عاطل عن العمل قبل قطع اتصاله الياُ.
CONCURRENT SESSIONS: لتحديد عدد الSESSEIONS المسموح بها للمستخدم للاتصال بقاعدة البيانات.
PRIVATE SGA: لتحديد المساحة المتاحة للمستخدم فى الPRIVATE SGA هذا إذا كنا نعمل فى الوضع SHARED SERVER.

الان نفترض الان اننا نريد أنشاء PROFILE جديد اسمه NEWPROFILE.

```
CREATE PROFILE NEWPROFILE LIMIT
FAILED_LOGIN_ATTEMPTS 3
PASSWORD_LOCK_TIME UNLIMITED
PASSWORD_LIFE_TIME 60
PASSWORD_REUSE_TIME 60
PASSWORD_GRACE_TIME 7
CONNECT_TIME 1000
IDLE_TIME 30
SESSIONS PER USER UNLIMITED;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> CREATE PROFILE NEWPROFILE LIMIT
  2  FAILED_LOGIN_ATTEMPTS 3
  3  PASSWORD_LOCK_TIME UNLIMITED
  4  PASSWORD_LIFE_TIME 60
  5  PASSWORD_REUSE_TIME 60
  6  PASSWORD_GRACE_TIME 7
  7  CONNECT_TIME 1000
  8  IDLE_TIME 30
  9  SESSIONS_PER_USER UNLIMITED;

Profile created.

SQL>
```

ماذا لو اردنا تعديل الPROFILE اعلاه

```
ALTER PROFILE NEWPROFILE LIMIT
SESSIONS_PER_USER 5;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> ALTER PROFILE NEWPROFILE LIMIT
  2  SESSIONS_PER_USER 5;

Profile altered.

SQL>
```

الان يمكن منح المستخدمين هذا الPROFILE.

```
ALTER USER TEST PROFILE NEWPROFILE;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> ALTER USER TEST PROFILE NEWPROFILE;

User altered.

SQL>
```

بالطبع يمكن منح هذا الPROFLE عند إنشاء المستخدم.

لعمل استلام عن الPROFILE
DBA_PROFILES

ولنفترض مثلاً اننا نريد معرفة تفاصيل PROFILE NEWPROFILE.

```
Command Prompt - SQLPLUS /NOLOG

SQL> SELECT RESOURCE_NAME,LIMIT FROM DBA_PROFILES WHERE PROFILE='NEWPROFILE';

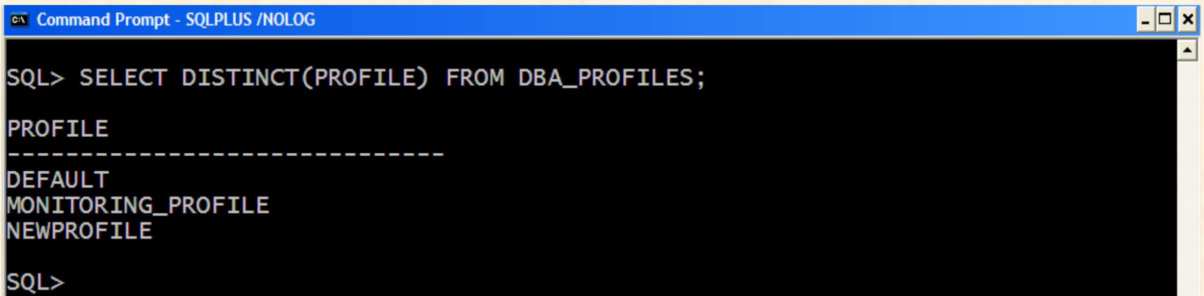
RESOURCE_NAME                    LIMIT
-------------------------------- ----------------------------------------
COMPOSITE_LIMIT                  DEFAULT
SESSIONS_PER_USER                5
CPU_PER_SESSION                  DEFAULT
CPU_PER_CALL                     DEFAULT
LOGICAL_READS_PER_SESSION        DEFAULT
LOGICAL_READS_PER_CALL           DEFAULT
IDLE_TIME                        30
CONNECT_TIME                     1000
PRIVATE_SGA                      DEFAULT
FAILED_LOGIN_ATTEMPTS            3
PASSWORD_LIFE_TIME               60

RESOURCE_NAME                    LIMIT
-------------------------------- ----------------------------------------
PASSWORD_REUSE_TIME              60
PASSWORD_REUSE_MAX               DEFAULT
PASSWORD_VERIFY_FUNCTION         DEFAULT
PASSWORD_LOCK_TIME               UNLIMITED
PASSWORD_GRACE_TIME              7

16 rows selected.
```

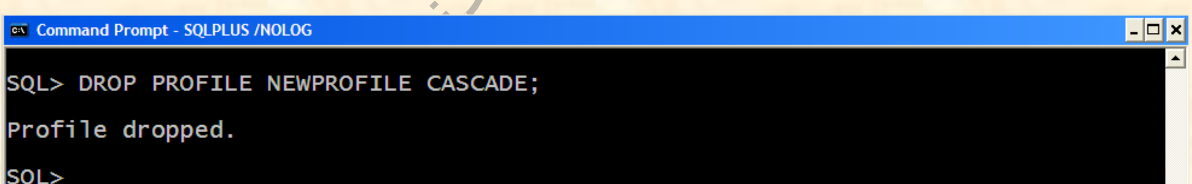
اذا اردنا معرفة كل الPROFILES الموجودة فى قاعدة البيانات

```
SELECT DISTINCT (PROFILE) FROM DBA_PROFILES;
```



كذلك يمكن حذف هذا الPROFILE

```
DROP PROFILE NEWPROFILE CASCADE;
```



الفصل السابع

تأمين ومراقبة قاعدة البيانات

Oracle Database Security & Monitoring:

فى الحقيقة هذا الفصل من الأهمية بمكان ، فالحفاظ على بقاء واتاحة قاعدة البيانات هو الامر الأهم لمدير قاعدة البيانات ، لذا كان لزاماً على مدير قاعدة البيانات تأمين قاعدة البيانات حتى لا تكون فى مهب الريح.

وحسب ملاحظتى فإن هذا الجانب لا يحظى بالاهتمام الاكبر بالنسبة لمديرى قاعدة البيانات بل يكون التركيز دائماً على وضع استراتيجيات مناسبة للنسخ الإحتياطى ، رغم أن أننا نلجأ كثيراً لإسترجاع النسخ الاحتياطية نتيجة تقصيرنا فى جانب تأمين قاعدة البيانات ، فلنرفع إذاً شعار الوقاية خير من العلاج ، ونبادر فى تأمين قاعدة البيانات .

الامر ليس معقداً لكن يعتمد على التخطيط ، خصوصا إذا كانت قاعدة البيانات ضخمة وعدد المستخدمين كبير .

والحديث فى هذا الفصل يمكن ان يقسم الى قسمين رئيسيين :-

1- تأمين قاعدة البيانات : وهو حماية قاعدة البيانات من المخاطر الداخلية والخارجية .

2- مراقبة قاعدة البيانات : فبعد المخاطر ناتجة عن سلوك بعض المستخدمين ، لذا كان على مدير قاعدة البيانات مراقبة ما يراه مهما لمتابعة سلوك المستخدين .

1- تأمين قاعدة البيانات :-

هناك الكثير من الخطوات التى يمكن أن يتخذها مدير قاعدة البيانات لتأمين فاعدة البيانات ، سأذكر منها ما اراه مهماً :

1- التأكد من تأمين الشبكة : اعرف أن الامر ليس من مهامك ادرك هذا جديداً لكن اجلس مع المسؤول من تأمين الشبكة وتأكد من إنتفاء جميع المخاطر ، لن تجد هذه النقطة فى كتب اوركل ؛ بل هى تجربة شخصية وانا اعمل فى مؤسسة مالية فى احد الدول العربية لولا أن النسخ الإحتياطى (Backup) كان حاضراً ؛ لكنا الان فى احد سجون الأمن الإقتصادى لهفوة قام بها المسؤول من تأمين الشبكة استطاع من خلالها احد القراصنة الدخول للمخدم وتعطيل قاعدة البيانات ، كنت واثقاً من عدم مسؤليتى من هذا الاختراق لكن مشكلتنا فى كثير من الدول العربية أننا لم نستطيع بعد استيعاب مفهوم التخصصية .

2- حماية الDATA DICTIONARY: كل المعلومات التي نحتاجها عن قاعدة البيانات موجودة فى الDATA DICTIONARY لذا كان لزاماً على مدير قاعدة البيانات حمايتها .
وذلك بوضع القيمة FALSE فى المتغير
.O7_DICTIONARY_ACCESSIBILITY

عموماً فى اوركل 10g هذا المتغير فى الاصل FALSE.
يمكن التأكد من ذلك عن طريق:

```
SHOW PARAMETER O7_DICTIONARY_ACCESSIBILITY;
```

```
SQL> SHOW PARAMETER O7_DICTIONARY_ACCESSIBILITY;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
O7_DICTIONARY_ACCESSIBILITY          boolean     FALSE
SQL>
```

بهذه التهيئة نضمن ان اى مستخدم يملك صلاحية الوصول الى كل جدول * ANY TABLE مثلاً DROP ANY TABLE لا يستطيع مسح الDATA DICTIONARY.
كذلك هذه التهيئة تمنع المستخدم SYS من الدخول لقاعدة البيانات بدون الصلاحية SYSDBA.

3- سحب الصلاحية الغير مهمة من PUBLIC :
وهو DATABASE SERVER USER GROP ، لان كل الصلاحيات التى تمنح لل PUBLIC يستطيع استخدمها مستخدمى قاعدة البيانات لذا يجب سحب كل الصلاحيات غير المهمة من الPUBLIC.

هكذا نستطيع منح الصلاحيات للPUBLIC.

```
GRANT ALTER ANY TABLE TO PUBLIC;
```

```
SQL> GRANT ALTER ANY TABLE TO PUBLIC;

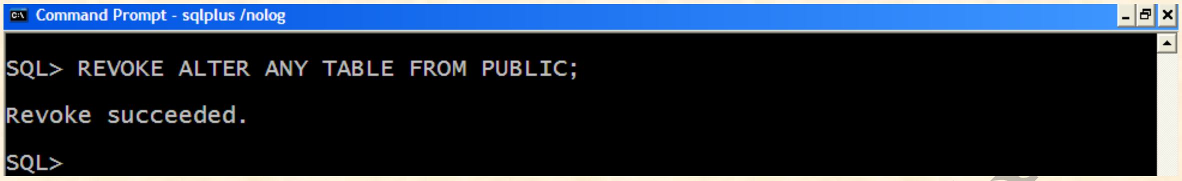
Grant succeeded.

SQL>
```

الان كل المستخدمين فى قاعدة البيانات يمكنهم استخدام هذه الصلاحية .

بالطبع يمكن سحبها.

```
REVOKE ALTER ANY TABLE FROM PUBLIC;
```



3- الحد من الصلاحيات الإدارية من المستخدميين : ما الفائدة من منح كل المستخدمين فى قاعدة البيانات DBA ROLE ، هذه الROLE تمنح لمدير قاعدة البيانات وليس لكل المستخدين ، كذلك الصلاحية SYSDBA & DROP AY TABLE وغيرها من الصلاحيات التى قد تسبب لك المشاكل ولا حاجة لمنحها لكل المستخدين .

4- منع التحقق من بعد عن طريق نظام التشغيل : وذلك بوضع القيمة FALSE فى المتغير REMOTE_OS_AUTHET ، حتى نمنع الاتصال عن بعد بقاعدة البيانات بواسطة التحقق عن طريق نظام التشغيل . هذا المتغير فى الاصل يأخذ القيمة FALSE فى اوركل 10g .
تحدثنا سابقاً عن طرق التحقق من المستخدين وذكرنا أن احد الطرق هو عن طريق نظام التشغيل EXTERNAL .

2- مراقبة قاعدة البيانات (MONITORING OR AUDITING):-

مراقبة قاعدة البيانات هو جزء من تأمين قاعدة البيانات ونستطيع من خلال عملية المراقبة ايجاد كثير من الاجابات لبعض الامور التى ستظل غامضة لولا عملية المراقبة ، ويمكن تقسيم عملية المراقبة الى ثلاث انواع :-

1- Standard Database Auditing: وهو لمتابعة وحفظ معلومات عن عمليات الإتصال وقطع الإتصال بقاعدة البيانات ، وايضاً متابعة أستخدام (System Privileges and Object Privileges) داخل قاعدة البيانات ، كعمليات الإنشاء والحذف والإستعلام والتعديل وغيره من العمليات .

2- Value-based auditing: فى النوع الاول تابعنا عمليات الإتصال وكذلك متابعة العمليات داخل قاعدة البيانات ، ولكن قد نحتاج للحفاظ عل القيم التى يتم تغيرها فى قاعدة البيانات .

3- Fine-grained auditing: هذا النوع لحفظ عبارات الSQL التى تم تنفيذها فى قاعدة البيانات .

ولكن قبل التفصيل فى انواع المراقبة لابد من الإشارة أن قاعدة البيانات اوركل 10g لابد من تهيئتها قبل بدء فى عمليات المراقبة بواسطة المتغير AUDIT_TRAIL الذى يأخذ أحد القيم التالية :-

1- NONE: وهى تعنى أن قاعدة البيانات غير جاهزة لعملية المراقبة .

2- DB: وهى تعنى ان معومات المراقبة يتم حفظها فى قاعدة البيانات فى جداول تنتمى للمستخدم SYS .

3- OS: وهى تعنى أن معلومات المراقبة يتم حفظها على مستوى نظام التشغيل ، فإذا كنا نعمل على بيئة ويندوز Widows فان البيانات تحفظ فى الEvent Log ، اما إذا كنا نعمل على UNIX or LINUX فإن المعلومات تخزن كملفات فى المسار المحدد على المتغير AUDIT_FILE_DEST .

الاصدار اوركل 10g يدعم مختلف انواع المراقبة ، بحيث يستطيع مدير قاعدة البيانات مراقبة كل الاحداث التى تحدث فى قاعدة البيانات ، تخيلت معى حجم

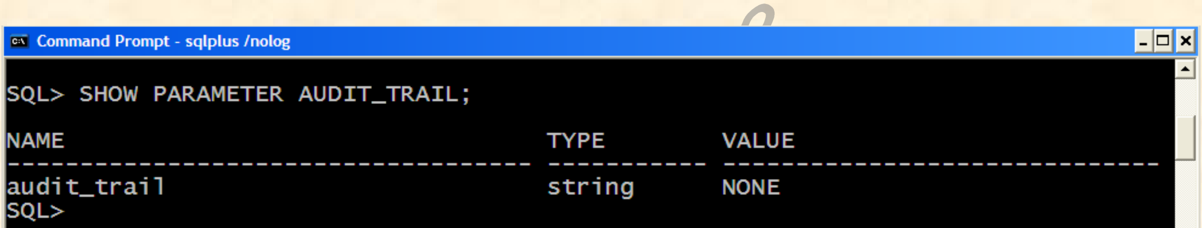
البيانات التى يتم تخزينها إذا قمنا بمراقبة كل ما يحدث فى قاعدة البيانات لا شك أن ذلك سينعكس سلباً على أداء قاعدة البيانات لذا كان لزاماً على مدير قاعدة البيانات التركيز فى المراقبة بحيث يتم مراقبة ما نحتاجه دون الافراط فى المراقبة .

فإذا أردنا مثلاً مراقبة بعض الجداول فالأفضل التركيز على تحديد العمليات التى يجب متابعتها مثلاً فقط الحذف ، اما اذا قمت بمتابعة كل ما يحدث للجداول فإن ذلك سيؤثر على اداء قاعدة البيانات فسوف يتم متابعة الإستعلام والتعديل والمسح وغيره ، إذا التركيز امر مهم فى عملية المراقبة .

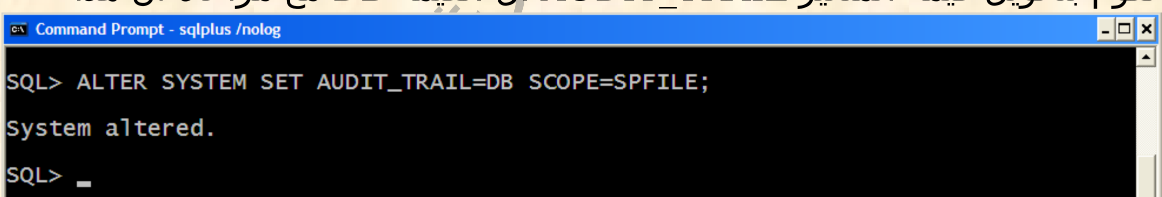
واول ما يلزمنا فعله هو تهيئة قاعدة البيانات بتغير قيمة المتغير AUDIT_TRAIL من القيمة NONE الى احد القيم التالية (DB OR OS) ولنفترض هنا أننا نريد تخزين معلومات المراقبة داخل قاعدة البيانات إذاً نختار القيمة DB.

لكن قبل ذلك نتأكد من المتغير AUDIT_TRAIL.

```
SHOW PARAMETER AUDIT_TRAIL;
```



نقوم بتحويل قيمة المتغير AUDIT_TRAIL ال القيمة DB مع مراعاة أن هذا



المتغير غير آلى ، أى يحتاج إلى إغلاق وفتح قاعدة البيانات حتى تتأثر قيمة المتغير .

```
ALTER SYSTEM SET AUDIT_TRAIL=DB SCOPE=SPFILE;
```

ثم نغلق ونفتح قاعدة البيانات ونتأكد من قيمة المتغير .

```
Command Prompt - sqlplus /nolog

SQL> SHUT IMMEDIATE;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> STARTUP;
ORACLE instance started.

Total System Global Area  171966464 bytes
Fixed Size                   787988 bytes
Variable Size             145488364 bytes
Database Buffers           25165824 bytes
Redo Buffers                 524288 bytes
Database mounted.
Database opened.
SQL> SHOW PARAMETER AUDIT_TRAIL;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
audit_trail                          string      DB
SQL> _
```


1- Standard Database Auditing:

كما ذكرنا سابقاً هذا النوع من المراقبة يتركز على مراقبة عملية الاتصال بقاعدة البيانات وكذلك متابعة إستخدام (System Privileges and Object Privileges) .

كما يجب الذكر أن هناك خيارين أثناء عملية المراقبة (BY SESSION & BY ACCESS) ،

BY SESSION: وهى تعنى تخزين حقل واحد من المعلومات أثناء عمليات المراقبة لنوع معين من العمليات لكل SESSION ، ولتضح الرؤية أكثر لنفترض

أننا قمنا بمراقبة عملية التعديل على جداول مستخدم معين ولنفترض أنه X ، فقام احد المستخدمين بفتح SESSION واستخدم هذه الSESSION لتعديل 5 جداول تنتمى للمستخدم X. فإن عملية المراقبة ستقوم بتخزين حقل واحد فقط لعملية التعديل ما دام أنه قام بالتعديل بنفس الSESSION ، اى بمعنى اخر أنها توازى عبارة GROUP BY SESSION.

BY ACCESS: يمكن اتخاذ نفس المثال السابق ، لكن هنا سيتم تخزين 5 حقول كل حقل هو عبارة عن عملية التعديل التى حدثت لكل جدول.

لا شك أن الخيار BY SESSION قد لا يكون كافياً احياناً لمعرفة تفاصيل معلومات المراقبة لذا قد نلجأ للخيار BY ACCESS الذى نجد فيه تفاصيل أكثر لكن بالطبع يزيد عمليات تخزين المعلومات.

فى هذا النوع من المراقبة كما ذكرنا الافضل فيه عملية التركيز ويكون التركيز بتحديد المستخدم كذلك بتحديد نجاح او فشل العملية (SUCCESSFUL OR NOT SUCCESSFUL) .

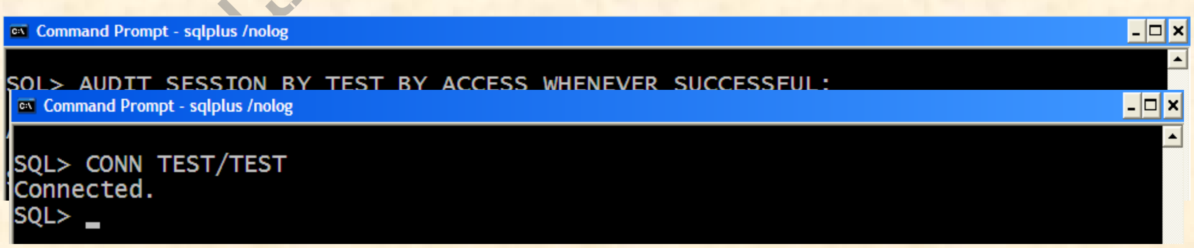
وهذا النوع من المراقبة Standard Database Auditing يحتوى على اقسام فرعية:-

*** SESSION AUDITING:**

وهى جزء من النوع الاول من المراقبة Standard Database Auditing ، بحيث يتم مراقبة عمليات الإتصال وقطع الإتصال بقاعدة البيانات ، ويمكن التركيز بتحديد المستخدم الذى نريد مراقبة عملياتة اتصاله كما يمكن متابعة كل المستخدمين بإستخدام الامر AUDIT SESSION كما يمكن التركيز ايضاً بتحديد عملية نجاح أو فشل عملية الإتصال .

نفترض أننا نريد مراقبة كل عمليات الإتصال الناجحة وقطع الإتصال بالنسبة للمستخدم TEST ولأننا نريد معرفة وقت الإتصال وقطع الإتصال فالأفضل استخدم الخيار BY ACCESS.

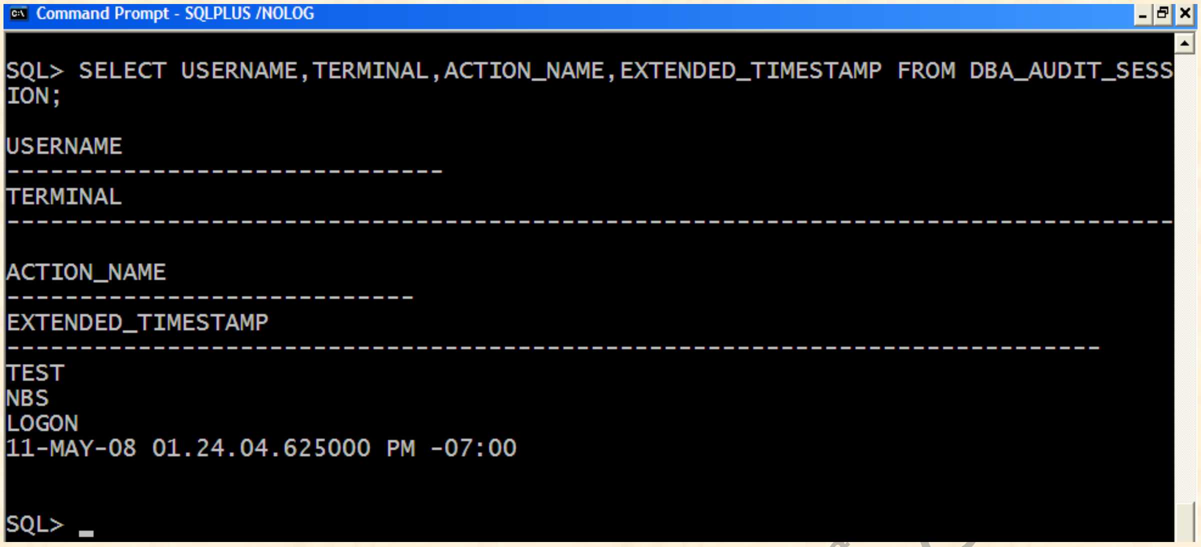
```
AUDIT SESSION BY TEST BY ACCESS WHENEVER
SUCCESSFUL;
```



ماذا الان لو قام المستخدم TEST بالاتصال بقاعدة البيانات

الان مدير قاعدة البيانات يستطيع أن يشاهد معلومات عن عملية الإتصال اعلاه

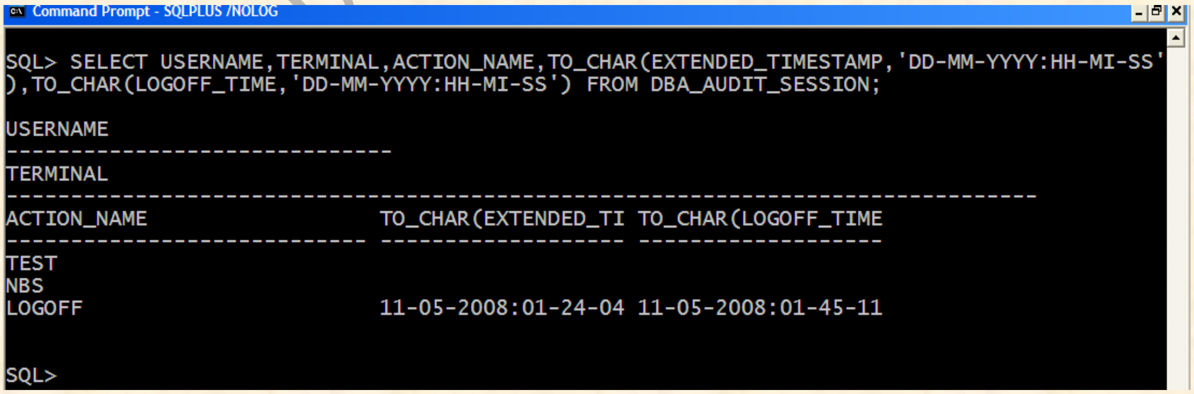
```
SELECT
SERNAME,TERMINAL,ACTION_NAME,EXTENDED_TIMESTAMP
FROM DBA_AUDIT_SESSION;
```



ظهر لمدير قاعدة البيانات ان المستخدم TEST قام بالإتصال بقاعدة البيانات بالتاريخ والزمن المحدد من الجهاز NBS كما يمكن استعراض معلومات اخرى عن اسم مستخدم الجهاز وغيره من المعلومات .

ماذا لو قام المستخدم TEST بقطع الإتصال ؟
بالطبع ستظهر لمدير قاعدة البيانات معلومات اضافية

```
SELECT
USERNAME,TERMINAL,ACTION_NAME,TO_CHAR(EXTENDED_TIMESTAMP,'DD-MM-YYYY:HH-MI-SS'),TO_CHAR(LOGOFF_TIME,'DD-MM-YYYY:HH-MI-SS')
FROM DBA_AUDIT_SESSION;
```



الان يظهر لمدير قاعدة البيانات الان المستخدم TEST قام بقطع الاتصال بقاعدة البيانات فى الزمن المحدد اعلاه ، اى أنه ظل متصل بقاعدة البيانات حوالى 21 دقيقة .

يمكن ايضاً استعلام معلومات الاتصال وقطع الاتصال بقاعدة البيانات بواسطة الجدول DBA_AUDIT_TRAIL .
إذاً معلومات الاتصال وقطع الإتصال بقاعدة البيانات متوفرة فى كل من (DBA_AUDIT_SESSION & DBA_AUDIT_TRAIL)

يمكن لمدير قاعدة البيانات الإستعلام عن خيارات المراقبة التى قام بتهيئتها فى قاعدة البيانات بواسطة
DBA_OBJ_AUDIT_OPTS
DBA_STMT_AUDIT_OPTS
DBA_PRIV_AUDIT_OPTS

```
SELECT USER_NAME,AUDIT_OPTION,SUCCESS,FAILURE FROM
```

```
SQL> SELECT USER_NAME,AUDIT_OPTION,SUCCESS,FAILURE FROM DBA_STMT_AUDIT_OPTS;

USER_NAME                      AUDIT_OPTION
------------------------------ ----------------------------------------
SUCCESS    FAILURE
---------- ----------
TEST                           CREATE SESSION
BY ACCESS  NOT SET


SQL>
```

بالطبع يمكن إلغاء عملية المراقبة ، فلو أردنا إلغاء عملية مراقبة إتصال المستخدم TEST التى قمنا بمراقبتها سابقاً.

```
NOAUDIT SESSION BY TEST;
```

```
SQL> NOAUDIT SESSION BY TEST;

Noaudit succeeded.

SQL>
```

الان لو قمنا بعملية استعلام لعرض خيارات المراقبة التى قمنا بتهيئتها فلن نجد خيار مراقبة اتصال المستخدم TEST.

```
SQL> SELECT USER_NAME,AUDIT_OPTION,SUCCESS,FAILURE FROM DBA_STMT_AUDIT_OPTS WHE
E='TEST';

no rows selected

SQL>
```

* SQL STATEMENT AUDITING:

تخيل معى أن مدير قاعدة البيانات يريد مراقبة عمليات الـDDL مثل إنشاء او حذف جدول . فى هذا النوع من المراقبة يمكن التركيز بواسطة تحديد المستخدم وكذلك بواسطة نجاح او فشل العملية ، لكن هذا فى إطار ما يملك المستخدم ، فلو قام مستخدم مثلاً بإنشاء جدول فى SCHEMA لمستخدم اخر فإن هذا النوع من المراقبة يدخل فيما يسمى SYSTEM PRIVILEGE AUDITING وهو ما سنناقشه لاحقا ، أما هنا فقط فى اطار ما يمكلك المستخدم فقط .

ولنفترض أننا نريد مراقبة عملية إنشاء الجداول بواسطة المستخدم TEST داخل مساحته اى فى نفس الـSCHEMA.

```
AUDIT CREATE TABLE BY TEST BY ACCESS;
```

```
Command Prompt - sqlplus /nolog
SQL> AUDIT CREATE TABLE BY TEST BY ACCESS;

Audit succeeded.

SQL>
```

ماذا لو قام الان المستخدم TEST بإنشاء جدول .

```
CREATE TABLE USER_MASTER (USER_NO
NUMBER(5),USER_NAME VARCHAR2(40),CONSTRAINT
PK_USER_NO PRIMARY KEY(USER_NO));
```

```
Command Prompt - SQLPLUS /NOLOG
SQL> CREATE TABLE USER_MASTER (USER_NO NUMBER(5),USER_NAME VARCHAR2(40),CONSTRAI
NT PK_USER_NO PRIMARY KEY(USER_NO));

Table created.

SQL>
```

ستظهر المعلومات لمدير قاعدة البيانات .

```
SELECT USERNAME,OWNER,TO_CHAR(TIMESTAMP,'DD-MM-
YYYY:HH-MI-SS'),OBJ_NAME,ACTION_NAME FROM
DBA_AUDIT_TRAIL WHERE OWNER='TEST';
```

```
Command Prompt - sqlplus /nolog

SQL> SELECT USERNAME,OWNER,TO_CHAR(TIMESTAMP,'DD-MM-YYYY:HH-MI-SS'),OBJ_NAME,ACT
ION_NAME FROM DBA_AUDIT_TRAIL WHERE OWNER='TEST';

USERNAME                       OWNER
------------------------------ ------------------------------
TO_CHAR(TIMESTAMP,'
-------------------
OBJ_NAME
--------------------------------------------------------------------------------

ACTION_NAME
----------------------------
TEST                           TEST
13-05-2008:02-25-43
USER_MASTER
CREATE TABLE


SQL>
```

لكن لو قام المستخدم TEST مثلاً بإنشاء جدول فى SCHEMA اخرى فإن خيارات المراقبة اعلاه لن تاتى بمعلومات اللهم إلا إذا رقبنا الصلاحية CREATE ANY TABLE وهو ما سنناقشه فى الخطوة القادمة .

يمكن الإستعلام عن الخيار التى قمنا بتهيئتها لمراقبة SQL STATEMENT بواسطة :

DBA_STMT_AUDIT_OPTS
DBA_PRIV_AUDIT_OPTS

```
Command Prompt - SQLPLUS /NOLOG

SQL> SELECT * FROM DBA_STMT_AUDIT_OPTS WHERE USER_NAME='TEST';

USER_NAME                      PROXY_NAME
------------------------------ ------------------------------
AUDIT_OPTION                             SUCCESS    FAILURE
---------------------------------------- ---------- ----------
TEST
CREATE TABLE                             BY ACCESS  BY ACCESS


SQL>
```

يمكن إلغاء المراقبة بواسطة

```
Command Prompt - SQLPLUS /NOLOG

SQL> NOAUDIT CREATE TABLE BY TEST;

Noaudit succeeded.

SQL>
```


*** SYSTEM PRIVILEGE AUDITING:**

وذلك لمراقبة عمليات استخدام الصلاحيات SYSTEM PRIVILEGES داخل قاعدة البيانات ، مثلاً الصلاحية CREATE ANT TABLE ، بالطبع يمكن تركيز عملية مراقبة الصلاحيات بواسطة مستخدم معين كذلك بواسطة نجاح العملية أو فشلها ، فى الاصل يتم تخزين عمليات مراقبة الصلاحيات SYSTEM PRIVILEGES كحقل لكل عملية وذلك لإستخدام الخيار BY ACCESS فى الاصل لكن بالطبع يمكن استعمال الخيار BY SESSION لتقليل عمليات تخزين المعلومات فيتم تخزين حقل واحد لكل SESSION قامت بإستخدام الصلاحية المراقبة .

ولنفترض أننا نريد مراقبة الصلاحية CREATE ANY TABLE للمستخدم TEST ، اى بمعنى اخر أننا نريد مراقبة كل الجداول التى يقوم المستخدم TEST بإنشاءها فى SCHEMA اخرى .
اى مستخدماً الصلاحية CREATE ANY TABLE وهى تعنى إنشاء جدول فى SCHEMA أخرى.

```
AUDIT CREATE ANY TABLE BY TEST WHENEVER
SUCCESSFUL;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> AUDIT CREATE ANY TABLE BY TEST WHENEVER SUCCESSFUL;

Audit succeeded.

SQL> _
```

الان ماذا لو قام المستخدم TEST بإنشاء جدول فى EMP SCHEMA اى فى مساحة المستخدم EMP.

```
CREATE TABLE EMP.EMPLOYEE (EMP_NO
NUMBER(4),EMP_NAME VARCHAR(40));
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> CREATE TABLE EMP.EMPLOYEE (EMP_NO NUMBER(4),EMP_NAME VARCHAR(40));

Table created.

SQL>
```

بالطبع يستطيع مدير قاعدة البيانات متابعة هذه الخطوة .

```
SELECT
USERNAME,OWNER,OBJ_NAME,ACTION_NAME,TO_CHAR(TIMEST
AMP,'DD-MM-YYYY:HH-MI-SS') FROM DBA_AUDIT_OBJECT
WHERE USERNAME='TEST';
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> SELECT USERNAME,OWNER,OBJ_NAME,ACTION_NAME,TO_CHAR(TIMESTAMP,'DD-MM-YYYY:HH
-MI-SS') FROM DBA_AUDIT_OBJECT WHERE USERNAME='TEST';

USERNAME                       OWNER
------------------------------ ------------------------------
OBJ_NAME
--------------------------------------------------------------------------------

ACTION_NAME                  TO_CHAR(TIMESTAMP,'
---------------------------- -------------------
TEST                           EMP
EMPLOYEE
CREATE TABLE                 12-05-2008:12-53-56


SQL> _
```

نعم لقد قام المستخدم TEST بإنشاء جدول EMPLOYEE فى المستخدم EMP فى الزمن المحدد اعلاه .
بالطبع يمكن متابعة باقى الصلاحيات مثلاً DROP ANY TABLE)) وغيرها من الصلاحيات SYSTEM PRIVILEGES.

للاستعلام عن تهيئة خيارات مراقبة ال SYSTEM PRIVILEGES.

```
SQL> SELECT * FROM DBA_PRIV_AUDIT_OPTS WHERE PRIVILEGE='CREATE ANY TABLE';

USER_NAME                      PROXY_NAME
------------------------------ ------------------------------
PRIVILEGE                                SUCCESS    FAILURE
---------------------------------------- ---------- ----------
TEST
CREATE ANY TABLE                         BY ACCESS  NOT SET


SQL>
```

يمكن إلغاء المراقبة

```
SQL> NOAUDIT CREATE ANY TABLE BY TEST WHENEVER SUCCESSFUL;

Noaudit succeeded.

SQL>
```

:OBJECT PRIVILEGE AUDITING *

وهذا النوع لمراقبة العمليات التى تحدث على الكائنات من جداول TABLES ومناظير VIEWS وغيرها من الكائنات .
يمكن التركيز بواسطة تحديد المستخدم وكذلك بواسطة نجاح أو فشل العملية .
الاصل فى هذا النوع من المراقبة أن تكون BY SESSION ولكن يمكن تحديد الخيار BY ACCESS.

لنفترض أننا نريد مراقبة عمليات الإضافة فى الجدول USER_MASTER الموجود فى الSCHEMA التى تسمى TEST.

```
AUDIT INSERT ON TEST.USER_MASTER BY ACCESS;
```

```
SQL> AUDIT INSERT ON TEST.USER_MASTER BY ACCESS;

Audit succeeded.

SQL>
```

الان لو قام المستخدم TEST بإضافة بيانات فى الجدول USER_MASTER .

```
SQL> INSERT INTO USER_MASTER VALUES(12,'MOHAMMED');

1 row created.

SQL> COMMIT;

Commit complete.

SQL>
```

يستطيع مدير قاعدة البيانات متابعة ذلك .

```
SELECT USERNAME,TO_CHAR(TIMESTAMP,'DD-MM-YY:HH-
MI-SS'),OBJ_NAME,ACTION_NAME FROM DBA_AUDIT_TRAIL
WHERE USERNAME='TEST' AND OBJ_NAME='USER_MASTER'
AND ACTION_NAME='INSERT';
```

```
SQL> SELECT USERNAME,TO_CHAR(TIMESTAMP,'DD-MM-YY:HH-MI-SS'),OBJ_NAME,ACTION_NAME
 FROM DBA_AUDIT_TRAIL WHERE USERNAME='TEST' AND OBJ_NAME='USER_MASTER' AND ACTIO
N_NAME='INSERT';

USERNAME                       TO_CHAR(TIMESTAMP
------------------------------ -----------------
OBJ_NAME
--------------------------------------------------------------------------------

ACTION_NAME
----------------------------
TEST                           13-05-08:05-31-16
USER_MASTER
INSERT


SQL>
```

للاستعلام عن خيارات تهيئة مراقبة ال OBJECT PRIVILEGES.

```
select * from DBA_OBJ_AUDIT_OPTS WHERE
OWNER='TEST' AND OBJECT_NAME='USER_MASTER';
```

```
SQL> select * from DBA_OBJ_AUDIT_OPTS WHERE OWNER='TEST' AND OBJECT_NAME='USER_M
ASTER';

OWNER                          OBJECT_NAME                    OBJECT_TY ALT AUD
------------------------------ ------------------------------ --------- --- ---
COM DEL GRA IND INS LOC REN SEL UPD REF EXE CRE REA WRI FBK
--- --- --- --- --- --- --- --- --- --- --- --- --- --- ---
TEST                           USER_MASTER                    TABLE     -/- -/-
-/- -/- -/- -/- A/A -/- -/- -/- -/- -/- -/- -/- -/- -/- -/-


SQL>
```

هناك ثلاث قيم :

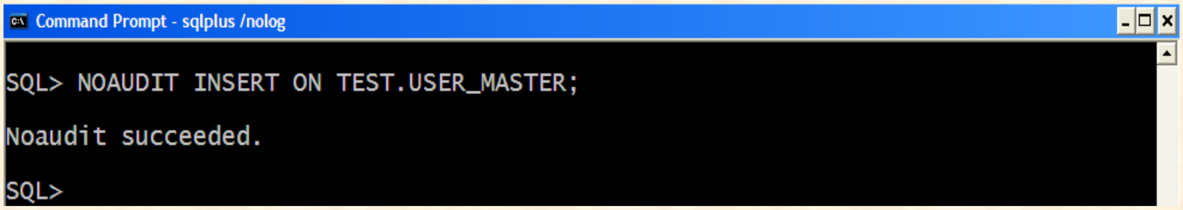
- : تعنى عدم المراقبة .

A: تعنى أن معلومات المراقبة تخزن BY ACCESS .

S: تعنى أن معلومات المراقبة تخزن BY SESSION.

لاحظ أنه مقابل الحقل INS وجدت القيمة A/A اى أنه تتم مراقبة عملية ال INSERT بواسطة الخيار BY ACCESS سواء نجحت العملية أو فشلت ، اما إذا كان الخيار WHENEVER SUCCESSFUL فإن القيمة ستكون A/-.

يمكن إلغاء عملية المراقبة بواسطة.



هكذا نكون قد ناقشنا الخيارات المتاحة للمراقبة التى تنتمى للجزء الاول
Standard Database Auditing وهى :
.SESSION AUDITING
SQL STATEMENT AUDITING
SYSTEM PRIVILEGE AUDITING
OBJECT PRIVILEGE AUDITING

:Value-Based Auditing -2

فى النوع الأول من المراقبة Standard Database Auditing
استطعنا مراقبة وحفظ معلومات عن العمليات التى تحدث فى قاعدة البيانات ، لكن قد نحتاج احياناً لحفظ القيم التى يتم تعديلها او حذفها فى قاعدة البيانات ، هذه القيم لا يتم حفظها فى الStandard Database Auditing .
يعمل هذا النوع من المراقبة Value-Based Auditing معتمداً على خلق Trigger بحيث يتم تنفيذه بعد عملية التعديل أو المسح ليحفظ المعلومات

والقيم التى نريدها ويحفظها فى جدول نقوم بخلقه خصيصاً لتخزين هذه المعلومات .

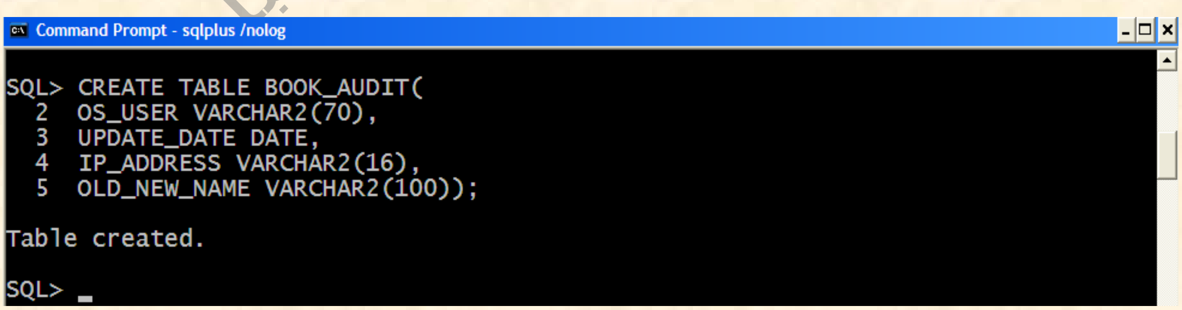
بالطبع هذا النوع من المراقبة Value-Based Auditing يقلل من أداء قاعدة البيانات اكثر من النوع Standard Database Auditing وذلك لأن ال Trigger يتم تنفيذه بعد كل عملية تعديل أو مسح .
ولنفترض الان أننا نريد مراقبة وحفظ القيم المعدلة فى الحقل BOOK_NAME الموجودة فى الجدول BOOK المملوك للمستخدم TEST.

يلزم مدير قاعدة البيانات اولاً تحديد المعلومات التى يريد حفظها ولنفترض أنها :-

1- IP Address للجهاز الذى استخدم للإتصال بقاعدة البيانات .
2- اسم مستخدم نظام التشغيل الذ قام بالإتصال بقاعدة البيانات .
3- التاريخ والزمن .
4- قيمة الحقل قبل التعدل وبعد التعديل .

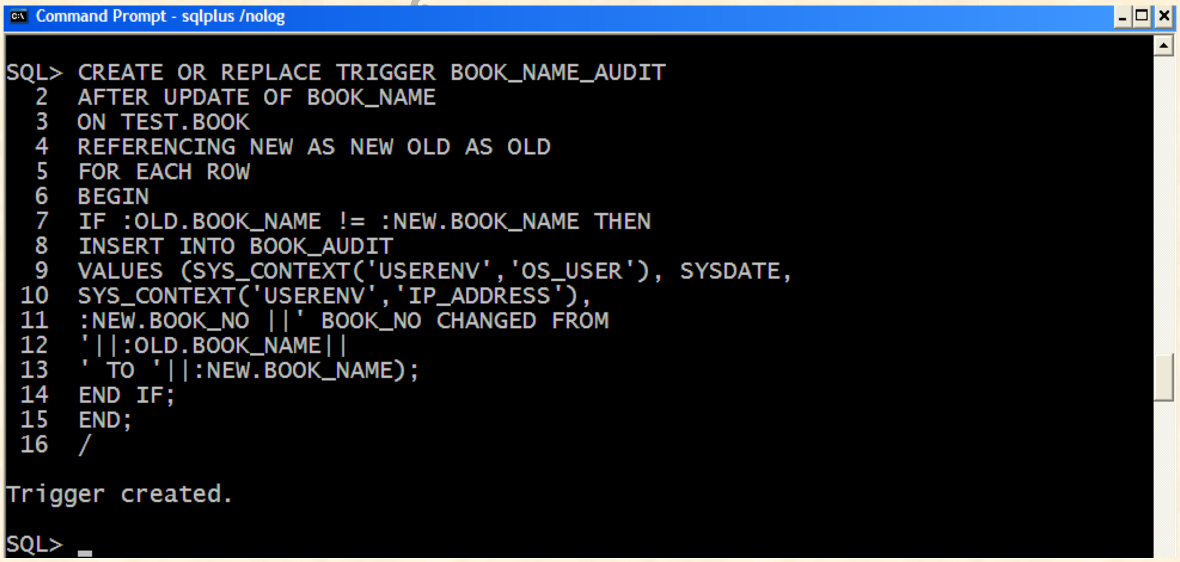
ثانياً: يلزم مدير قاعدة البيانات إنشاء الجدول لتخزين المعلومات عن عمليات التعديل ,

```
CREATE TABLE BOOK_AUDIT(
OS_USER VARCHAR2(70),
UPDATE_DATE DATE,
IP_ADDRESS VARCHAR2(16),
OLD_NEW_NAME VARCHAR2(100));
```



ثالثاً يقوم مدير قاعدة البيانات بإنشاء الTrigger.

```
CREATE OR REPLACE TRIGGER BOOK_NAME_AUDIT
AFTER UPDATE OF BOOK_NAME
ON TEST.BOOK
REFERENCING NEW AS NEW OLD AS OLD
FOR EACH ROW
BEGIN
IF :OLD.BOOK_NAME != :NEW.BOOK_NAME THEN
INSERT INTO BOOK_AUDIT
VALUES (SYS_CONTEXT('USERENV','OS_USER'),
SYSDATE,
SYS_CONTEXT('USERENV','IP_ADDRESS'),
:NEW.BOOK_NO ||' BOOK_NO( CHANGED FROM
'||:OLD.BOOK_NAME||
' TO '||:NEW.BOOK_NAME);
END IF;
END;
/
```



هكذا تم إنشاء الTrigger ولنفترض الان أن المستخدم TEST قام بتعديل فى

```
SQL> UPDATE BOOK SET BOOK_NAME='ORACLE10g' WHERE BOOK_NO=1;

1 row updated.

SQL> commit;

Commit complete.

SQL> _
```

الجدول BOOK.

يستطيع مدير قاعدة البيانات متابعة ذلك التعديل بواسطة الجدول BOOK_AUDIT.

لايقاف هذ النوع من المراقبة يمكن تعطيل أو حذف هذا الTrigger.

3- Fine-Grained Auditing (FGA):

الأنواع السابقة من المراقبة تستطيع حفظ معلومات عن العمليات التى تحدث فى قاعدة البيانات كما يمكن حفظ القيم التى تتغير ، أما هذا النوع من المراقبة فهو لحفظ عبارات الSQL التى تنفذ فى قاعدة البيانات ويشمل كل من العبارات الاتية : (SELECT & INSERT & UPDATE & DELETE) وذلك باستخدام DBMS_FGA PACKAGE ، هذه الحزمة تحتوى على اربعة من ال

```
SELECT OS_USER,TO_CHAR(UPDATE_DATE,'DD-MM-YY:HH-
MI-SS'),IP_ADDRESS,OLD_NEW_NAME FROM BOOK_AUDIT;
```

```
SQL> SELECT OS_USER,TO_CHAR(UPDATE_DATE,'DD-MM-YY:HH-MI-SS'),IP_ADDRESS,OLD_NEW_
NAME FROM BOOK_AUDIT;

OS_USER
----------------------------------------------------------------------
TO_CHAR(UPDATE_DA IP_ADDRESS
----------------- ----------------
OLD_NEW_NAME
--------------------------------------------------------------------------------

NBS\Administrator
14-05-08:04-40-49
1 BOOK_NO CHANGED FROM
PLSQL TO ORACLE10g

SQL> _
```

Procedure :-

1- ADD_POLICY: لإضافة POLICY جديدة لمراقبة عبارات الSQL فى قاعدة البيانات وذلك حسب القيم التى سنحددها فى الإجراء .

2- PROP_POLICY: لحذف POLICY AUDIT موجودة.

3- ENABLE_POLICY: لتشغيل POLICY AUDIT كانت معطلة .

4- DISABLE_POLICY: لتعطيل عمل POLICY AUDIT.

ولنفترض الان أننا نريد حفظ عبارات الSELECT التى تحدث للجدول BOOK المملوك للمستخدم TEST . وحتى لا نحفظ كل عبارات الSELECT التى تحدث للجدول BOOK فمن الممكن التركيز على بعض الحقول وذلك بإستخدام بعض الشروط .
مثلاً WHERE BOOK_NO=1.
وذلك لتركيز عملية المراقبة .

مدير قاعدة البيانات دائماً يجب ان يراعى أن لا يفرط فى عملية المراقبة ، بل يركز دائما على المطلوب .
الان نقوم بتنفيذ الإجراء ADD_POLICY وذلك لإضافة AUDIT POLICY تقوم بحفظ عبارات الSELECT التى تنفذ على الجدول BOOK وذلك حسب الشروط التى نحددها .

بالطبع هذا الإجراء Procedure يحتوى على مجموعة من المتغيرات يجب أن نحدد لها قيم ، هذه المتغيرات سنتحدث عنها بشى من التفصيل لاحقاً .

هذه الإجراءات Procedures موجودة فى قاعدة البيانات داخل الحزمة DBMG_FGA اى لا تحتاج منك انشاؤها فقط تحتاج لتنفيذها بعد تحديد قيم المتغيرات .

```
DECLARE
  OBJECT_SCHEMA VARCHAR2(200);
  OBJECT_NAME VARCHAR2(200);
  POLICY_NAME VARCHAR2(200);
  AUDIT_CONDITION VARCHAR2(200);
  AUDIT_COLUMN VARCHAR2(200);
  HANDLER_SCHEMA VARCHAR2(200);
  HANDLER_MODULE VARCHAR2(200);
  ENABLE BOOLEAN;
  STATEMENT_TYPES VARCHAR2(200);
  AUDIT_TRAIL BINARY_INTEGER;
  AUDIT_COLUMN_OPTS BINARY_INTEGER;

BEGIN
  OBJECT_SCHEMA := 'TEST';
  OBJECT_NAME := 'BOOK';
  POLICY_NAME := 'BOOK_SELEC';
  AUDIT_CONDITION := 'BOO_NO=1';
  AUDIT_COLUMN := 'BOOK_NAME';
  HANDLER_SCHEMA := NULL;
  HANDLER_MODULE := NULL;
  ENABLE := TRUE;
  STATEMENT_TYPES := 'SELECT';
  AUDIT_TRAIL := 1;
  AUDIT_COLUMN_OPTS := 0;

  SYS.DBMS_FGA.ADD_POLICY ( OBJECT_SCHEMA,
OBJECT_NAME, POLICY_NAME, AUDIT_CONDITION,
AUDIT_COLUMN, HANDLER_SCHEMA, HANDLER_MODULE, ENABLE,
STATEMENT_TYPES, AUDIT_TRAIL, AUDIT_COLUMN_OPTS );
  COMMIT;
END;
/
```

OBJECT_SCHEMA: وهى تعنى المستخدم الذى يحتوى الجدول او الكائن الذ نريد مراقبة عبارات الSELECT عليه.

OBJECT_NAME: وهو الكائن الذى نريد مراقبة عبارات الSELECT عليه.

POLICY_NAME: وهو اسم الAUDIT POLICY التى نريد خلقها.

AUDIT_CONDITION: وهى لتحديد الشروط لتركيز عملية المراقبة.

AUDIT_COLUMN: وهو العمود الذى نريد تركيز عملية المراقبة عليه.

HANDLER_SCHEMA: احياناً قد نحتاج لتحديد اجراء Procedure ليقوم ببعض الاحداث الإضافية اثناء عملية المراقبة ، فنقوم هنا بتحديد المستخدم الذى يحوى هذا الإجراء.

ENABLE : لتفعيل عمل AUDIT POLICY ويأخذ هذا المتغير القيمة TRUE فى الاصل.

STATEMENT_TYPES : لتحديد نوع عبارات الSQL التى نريد مراقبتها والاصل SELECT.

AUDIT_TRAIL: لتخزين عبارات الSQL المراقبة فى الAUDIT_TRAIL.

AUDIT_COLUMN_OPTS: ماذا لو قمت بتحديد عدد من الاعمدة فى المتغير AUDIT_COLUMN ، هنا لتحديد هل (ANY OR ANY) الاصل يحتوى عل القيمة 0 وهى تعنى ANY.

```
SQL> DECLARE
  2    OBJECT_SCHEMA VARCHAR2(200);
  3    OBJECT_NAME VARCHAR2(200);
  4    POLICY_NAME VARCHAR2(200);
  5    AUDIT_CONDITION VARCHAR2(200);
  6    AUDIT_COLUMN VARCHAR2(200);
  7    HANDLER_SCHEMA VARCHAR2(200);
  8    HANDLER_MODULE VARCHAR2(200);
  9    ENABLE BOOLEAN;
 10    STATEMENT_TYPES VARCHAR2(200);
 11    AUDIT_TRAIL BINARY_INTEGER;
 12    AUDIT_COLUMN_OPTS BINARY_INTEGER;
 13
 14  BEGIN
 15    OBJECT_SCHEMA := 'TEST';
 16    OBJECT_NAME := 'BOOK';
 17    POLICY_NAME := 'BOOK_SELECT';
 18    AUDIT_CONDITION := 'BOOK_NO=1';
 19    AUDIT_COLUMN := 'BOOK_NAME';
 20    HANDLER_SCHEMA := NULL;
 21    HANDLER_MODULE := NULL;
 22    ENABLE := TRUE;
 23    STATEMENT_TYPES := 'SELECT';
 24    AUDIT_TRAIL := 1;
 25    AUDIT_COLUMN_OPTS := 0;
 26
 27    SYS.DBMS_FGA.ADD_POLICY ( OBJECT_SCHEMA, OBJECT_NAME, POLICY_NAME, AUDIT_
CONDITION, AUDIT_COLUMN, HANDLER_SCHEMA, HANDLER_MODULE, ENABLE, STATEMENT_TYPES
, AUDIT_TRAIL, AUDIT_COLUMN_OPTS );
 28    COMMIT;
 29  END;
 30  /

PL/SQL procedure successfully completed.
```

الان ماذا لو قام المستخدم EMP بعمل استعلام على الجدول BOOK المنتمى للمستخدم TEST بنفس الشروط المذكورة اعلاه ؟

```
SELECT * FROM TEST.BOOK WHERE BOOK_NO=1;
```

```
Command Prompt - sqlplus /nolog

SQL> SELECT * FROM TEST.BOOK WHERE BOOK_NO=1;

   BOOK_NO BOOK_NAME
---------- -------------------------------------------------------
         1 ORACLE10g

SQL>
```

يستطيع مدير قاعدة البيانات الان متابعة هذا الإستعلام.

```
SELECT
USERHOST,OS_USER,DB_USER,TO_CHAR(TIMESTAMP,'DD-MM-
YY:HH-MI-SS'),SQL_TEXT FROM DBA_FGA_AUDIT_TRAIL
WHERE DB_USER='EMP;
```

```
Command Prompt - sqlplus /nolog

SQL> SELECT USERHOST,OS_USER,DB_USER,TO_CHAR(TIMESTAMP,'DD-MM-YY:HH-MI-SS'),SQL_
TEXT FROM DBA_FGA_AUDIT_TRAIL WHERE DB_USER='EMP';

USERHOST
--------------------------------------------------------------------------------

OS_USER
--------------------------------------------------------------------------------

DB_USER                        TO_CHAR(TIMESTAMP
------------------------------ -----------------
SQL_TEXT
--------------------------------------------------------------------------------

WORKGROP\NBS
NBS\Administrator
EMP                            15-05-08:12-44-04
SELECT * FROM TEST.BOOK WHERE BOOK_NO=1


SQL>
```

لو قام المستخدم EMP بعمل إستعلام اخر لم تتحق فيه الشروط الموضحة فى الإجراء ADD_POLICY فإن معلومات هذا الإستعلام لن تظهر لمدير قاعدة البيانات.

بالطبع يستطيع مدير قاعدة البيانات الاستعلام عن AUDIT_POLICIES التى تعمل فى قاعدة البيانات .
.DBA_AUDIT_POLICIES

```
SELECT
POLICY_NAME,OBJECT_SCHEMA,OBJECT_NAME,POLICY_COLUM
N,SEL FROM DBA AUDIT POLICIES;
```

Command Prompt - sqlplus /nolog

```
SQL> SELECT POLICY_NAME,OBJECT_SCHEMA,OBJECT_NAME,POLICY_COLUMN,SEL FROM DBA_AUD
IT_POLICIES;

POLICY_NAME                    OBJECT_SCHEMA
------------------------------ ------------------------------
OBJECT_NAME                    POLICY_COLUMN                  SEL
------------------------------ ------------------------------ ---
BOOK_SELECT                    TEST
BOOK                           BOOK_NAME                      YES


SQL>
```

يمكن تعطيل هذه الAUDIT POLICY بواسطة الإجراء DISABLE_POLICY.

Command Prompt - sqlplus /nolog

```
DECLARE
  OBJECT_SCHEMA VARCHAR2(200);
  OBJECT_NAME VARCHAR2(200);
  POLICY_NAME VARCHAR2(200);

BEGIN
  OBJECT_SCHEMA := 'TEST';
  OBJECT_NAME := 'BOOK';
  POLICY_NAME := 'BOOK_SELECT';

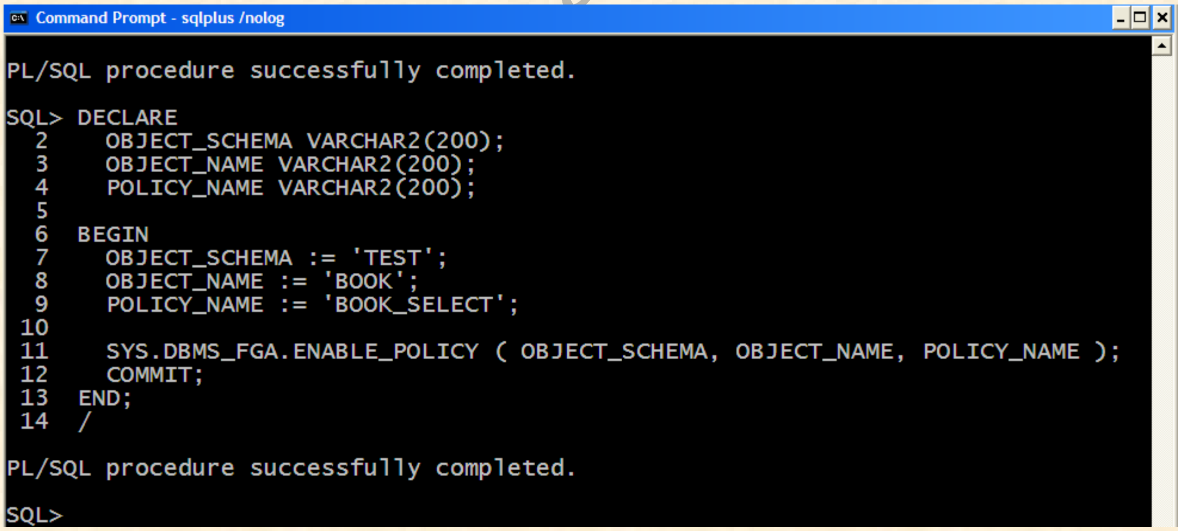
  SYS.DBMS_FGA.DISABLE_POLICY ( OBJECT_SCHEMA,
OBJECT_NAME, POLICY_NAME );
  COMMIT;
END;
/
```

كما يمكن تفعيلها مره اخرى بواسطة بواسطة الاجراء ENABLE_POLICY.

```
DECLARE
  OBJECT_SCHEMA VARCHAR2(200);
  OBJECT_NAME VARCHAR2(200);
  POLICY_NAME VARCHAR2(200);

BEGIN
  OBJECT_SCHEMA := 'TEST';
  OBJECT_NAME := 'BOOK';
  POLICY_NAME := 'BOOK_SELECT';

  SYS.DBMS_FGA.ENABLE_POLICY ( OBJECT_SCHEMA,
OBJECT_NAME, POLICY_NAME );
  COMMIT;
END;
/
```


```
PL/SQL procedure successfully completed.

SQL> DECLARE
  2    OBJECT_SCHEMA VARCHAR2(200);
  3    OBJECT_NAME VARCHAR2(200);
  4    POLICY_NAME VARCHAR2(200);
  5
  6  BEGIN
  7    OBJECT_SCHEMA := 'TEST';
  8    OBJECT_NAME := 'BOOK';
  9    POLICY_NAME := 'BOOK_SELECT';
 10
 11    SYS.DBMS_FGA.ENABLE_POLICY ( OBJECT_SCHEMA, OBJECT_NAME, POLICY_NAME );
 12    COMMIT;
 13  END;
 14  /

PL/SQL procedure successfully completed.

SQL>
```

واخيراً يمكن حذف AUDIT POLICY بواسطة الإجراء DROP_POLICY.

```
DECLARE
  OBJECT_SCHEMA VARCHAR2(200);
  OBJECT_NAME VARCHAR2(200);
  POLICY_NAME VARCHAR2(200);

BEGIN
  OBJECT_SCHEMA := 'TEST';
  OBJECT_NAME := 'BOOK';
  POLICY_NAME := 'BOOK_SELECT';

  SYS.DBMS_FGA.DROP_POLICY ( OBJECT_SCHEMA,
OBJECT_NAME, POLICY_NAME );
  COMMIT;
END;
/
```

```
Command Prompt - sqlplus /nolog

SQL> DECLARE
  2    OBJECT_SCHEMA VARCHAR2(200);
  3    OBJECT_NAME VARCHAR2(200);
  4    POLICY_NAME VARCHAR2(200);
  5
  6  BEGIN
  7    OBJECT_SCHEMA := 'TEST';
  8    OBJECT_NAME := 'BOOK';
  9    POLICY_NAME := 'BOOK_SELECT';
 10
 11    SYS.DBMS_FGA.DROP_POLICY ( OBJECT_SCHEMA, OBJECT_NAME, POLICY_NAME );
 12    COMMIT;
 13  END;
 14  /

PL/SQL procedure successfully completed.

SQL>
```

الان لو قمت بعمل إستعلام عن الAUDIT_POLICIES فلن تجد هذه ال POLICY فى قاعدة البيانات

الفصل الثامن

:: Oracle Net Services & Database Link & metrlized View

1- Oracle Net Services :-

أن يكون لديك مخدم Database Server يحتوى على قاعدة البيانات يتصل به جميع Client Application او جميع الاجهزة التى تحتوى على برامج تحتاج للاتصال بقاعدة البيانات هذا هو الامر الطبيعى والمعتاد والذى تعمل به اغلب الشركات فى العالم ، إذ لا يتصور أن تكون جميع الاجهزة التى تحتوى على برامج تحتاج للاتصال بقاعدة البيانات تحتوى ايضاً على قاعدة البيانات ، وإلا فإننا نحتاج لقاعدة البيانات لكل جهاز يحتوى عل برنامج وهذا غير مفبول عقلاً ولا عملاً.

وإذا سلمنا بأن يكون لدينا مخدم يحتوى على قاعدة البيانات Database Server تستطيع جميع الاجهزة التى تحتوى على برامج تحتاج لقاعدة البيانات الاتصال بهذا المخدم ، فإن محور حديثنا فى هذا الفصل سينصب على كيفية إنجاح هذا الاتصال .

إذاً الامر سيكون على جانبين : الجانب الاول وهو جانب المخدم Database Server وكيف يستطيع خدمة جميع الطلبات التى تصله للاتصال بقاعدة البيانات .
اما الجانب الثانى وهو جانب الClient Application وهو الجهاز الذى يريد الإتصال بقاعدة البيانات ، وكيف يستطيع الوصول لقاعدة البيانات ؟

والان سنتحدث عن الجانب الاول وهو جانب المخدم Database Server وكيف يستطيع خدمة جميع الطلبات للاتصال بقاعدة البيانات ؟

فى هذا الجانب يستطيع المخدم Database Server خدمة طلبات الاتصال بقاعدة البيانات بواسطة الOracle Net Listener وهو المسؤول عن عملية معالجة طلبات الClients للاتصال بقاعدة البيانات ، فبدون الListener محاولة عملية الاتصال بقاعدة البيانات من خارج المخدم ستبوء بالفشل ، لكن عملية محاولة الإتصال بقاعدة البيانات من داخل المخدم لا تحتاج للListener إذ أنه متخصص لإستقبال الطلبات الخارجية .

يستطيع مستمع واحد One Listener خدمة عدد من ال Database Instances ، وهو فى الاصل عبار عن ملف يوجد فى المسار الاتى إذا كنا نعمل على نظام التشغيل WINDOWS.
%ORACLE_HOME%\NETWORK\ADMIN\LISTENER.ORA
أما إذا كنا نعمل على نظام التشغيل UNIX
$ORACLE_HOME/NETWORK/ADMIN/LISTENER.ORA

يستطيع مدير قاعدة البيانات تحرير وتهيئة هذا الملف ليستقبل طلبات الاتصال بقاعدة البيانات ،

```
listener.ora - Notepad
File  Edit  Format  View  Help
# listener.ora Network Configuration File: D:\oracle\product\10.1.0\db_1\NETWORK\ADMIN\listener.ora
# Generated by Oracle configuration tools.

SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = D:\oracle\product\10.1.0\db_1)
      (PROGRAM = extproc)
    )
    (SID_DESC =
      (GLOBAL_DBNAME = ORCL)
      (SID_NAME = ORCL)
    )
  )

LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC))
    )
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = TCP)(HOST = nbs)(PORT = 1521))
    )
  )
```

هذا نموذج لملف الListener بحيث يحتوى على One Listener يسمى Listener يعمل فى المخدم nbs ويراقب الPort 1521 مستخدماً TCP PROTOCOL ، هذا المستمع يخدم INSTANCE تسمى ORCL.

بالطبع يستطيع مدير قاعدة البيانات إضافة مستمع LISTENER جديد ولنفترض أننا نريد إضافة مستمع جديد يسمى LISTENER1 يعمل فى نفس المخدم NBS يراقب الPORT 1521 ويخدم نفس الINSTANCE التى تسمى ORCL مستخدماً TCP PROTOCOL .

يكون شكل الملف بعد التعديل كالاتى :-

```
# listener.ora Network Configuration File: D:\oracle\product\10.1.0\db_1\NETWORK\ADMIN\listener.ora
# Generated by Oracle configuration tools.

LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC))
    )
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = TCP)(HOST = nbs)(PORT = 1521))
    )
  )
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = D:\oracle\product\10.1.0\db_1)
      (PROGRAM = extproc)
    )
    (SID_DESC =
      (GLOBAL_DBNAME = ORCL)
      (SID_NAME = ORCL)
    )
  )
LISTENER1 =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC))
    )
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = TCP)(HOST = nbs)(PORT = 1521))
    )
  )
SID_LIST_LISTENER1 =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = D:\oracle\product\10.1.0\db_1)
      (PROGRAM = extproc)
    )
    (SID_DESC =
      (GLOBAL_DBNAME = ORCL)
      (SID_NAME = ORCL)
    )
  )
```

لاحظت التعديلات الجديدة فى الملف : وهى إضافة مستمع جديد يسمى .LISTENER1

بعد ذلك نستطيع التحكم فى LISTENERS بواسطة الامر .LSNRCTL

```
Command Prompt - LSNRCTL

C:\Documents and Settings\Administrator>LSNRCTL

LSNRCTL for 32-bit Windows: Version 10.1.0.2.0 - Production on 26-MAY-2008 23:09
:01

Copyright (c) 1991, 2004, Oracle.  All rights reserved.

Welcome to LSNRCTL, type "help" for information.

LSNRCTL> HELP
The following operations are available
An asterisk (*) denotes a modifier or extended command:

start               stop                status
services            version             reload
save_config         trace               change_password
quit                exit                set*
show*

LSNRCTL>
```

الاوامر المعروضة ه المتاحة للتعامل مع الLISTENER.
وهى اوامر لتشغيل وايقاف وعرض حالة المستمع وكذلك وضع كلمة سر وتغيرها للمستمع كنوع من التحقق ، وكذلك إعادة تشغيل المستمع ليستوعب ما تم تحديثه من تهيئة لملف الLISTERNER.ORA وغيره من الاوامر ، وبما أننا اضفنا مستمع جديد اسميناه LISTENER1 فالافضل أن نقوم بتشغيل هذا المستمع ، لكن يجب التنبيه إلى أنه عند كتابة الامر LISTENER فإنه يتم التعامل مع المستمع الاصل اى DEFAULT والذى هنا هو LISTENER لذا إذا اردنا أن نتعامل مع المستمع غير الاصلى فامامنا احد خياران:
الاول : تحيد إسم المستمع عند توجيه الامر

```
LSNRCTL> START LISTENER1
Starting tnslsnr: please wait...

TNSLSNR for 32-bit Windows: Version 10.1.0.2.0 - Production
System parameter file is D:\oracle\product\10.1.0\db_1\network\admin\listener.or
a
Log messages written to D:\oracle\product\10.1.0\db_1\network\log\listener1.log
Listening on: (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(PIPENAME=\\.\pipe\EXTPROCipc)
))
Listening on: (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=nbs)(PORT=1521)))

Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=IPC)(KEY=EXTPROC)))
STATUS of the LISTENER
------------------------
Alias                     LISTENER1
Version                   TNSLSNR for 32-bit Windows: Version 10.1.0.2.0 - Produ
ction
Start Date                26-MAY-2008 23:23:40
Uptime                    0 days 0 hr. 0 min. 2 sec
Trace Level               off
Security                  ON: Local OS Authentication
SNMP                      OFF
Listener Parameter File   D:\oracle\product\10.1.0\db_1\network\admin\listener.o
ra
Listener Log File         D:\oracle\product\10.1.0\db_1\network\log\listener1.lo
g
Listening Endpoints Summary...
  (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(PIPENAME=\\.\pipe\EXTPROCipc)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=nbs)(PORT=1521)))
Services Summary...
Service "ORCL" has 1 instance(s).
  Instance "ORCL", status UNKNOWN, has 1 handler(s) for this service...
Service "PLSExtProc" has 1 instance(s).
  Instance "PLSExtProc", status UNKNOWN, has 1 handler(s) for this service...
The command completed successfully
```

الان قمنا بتشغيل المستمع الجديد الذى انشأناه وهو المستمع LISTENER1 ، ولو لم نكتب إسم المستمع بعد الامر START لتم التعامل مع المستمع الاصلى وهو المستمع LISTENER.

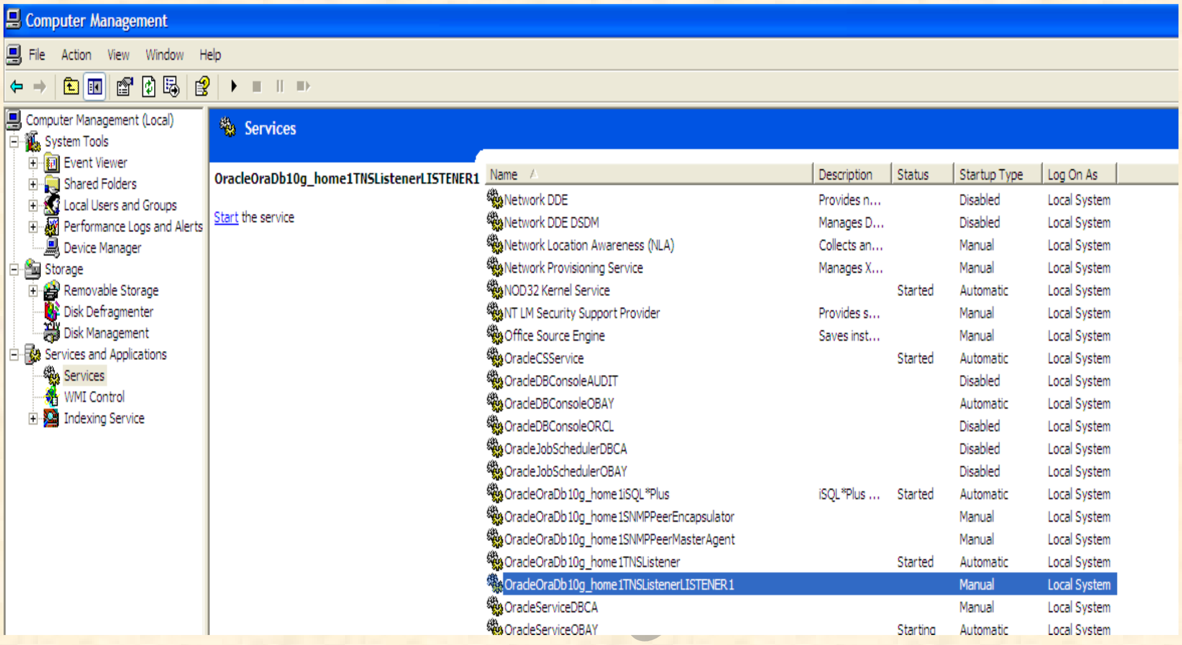
الخيار الثانى : استعمال الامر SET CUR LISTENER_NAME لوضع ال DEFAULT LISTENER الجديد ، اى سيصبح الاسم الجديد للمستمع هو الاصل الذى سنتعامل معه .

```
Command Prompt - LSNRCTL

LSNRCTL> SET CUR LISTENER1
Current Listener is LISTENER1
LSNRCTL> _
```

الان LISTENER1 هو الDEFAULT LISTENER.

بعد إنشاء المستمع LISTENER1 وتشغيله يمكن متابعته عن طريق ال SERVICES إذا كنا نعمل عل نظام التشغيل WINDOWS.



الان المستمع LISTENER1 يستطيع استقبال طلبات الإتصال بقاعدة البيانات على ال1521 PORT فى المخدم NBS مستعملاً TCP PROTOCOL ليخدم ORCL INSTANCE.

إذا أردنا مراقبة ومعرفة معلومات عن المستمع فالافضل استخدام الامر STATUS او SERVICE لعرض معلومات عن المستمع مثل اسم المستمع واصداره ومتى تم تشغيله والservices التى يخدمها ، ومسار ملف ال LISTENER.ORA وغيرها من المعلومات.

كما ذكرنا أن مستمع واحد يستطيع خدمة عدد من الINSTANCES كما يمكن ان يتشارك عدد من المستمعين LISTENERS فى خدمة INSTANCE واحدة ، كما يمكن ان يكون هناك عدد من الINSTANCES فى الجهاز الواحد فيكون لكل واحد منها مستع يخدمها .

```
LSNRCTL> STATUS
Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=IPC)(KEY=EXTPROC)))
STATUS of the LISTENER
------------------------
Alias                     LISTENER
Version                   TNSLSNR for 32-bit Windows: Version 10.1.0.2.0 - Produ
ction
Start Date                26-MAY-2008 23:57:51
Uptime                    0 days 0 hr. 17 min. 18 sec
Trace Level               off
Security                  ON: Local OS Authentication
SNMP                      OFF
Listener Parameter File   D:\oracle\product\10.1.0\db_1\network\admin\listener.o
ra
Listener Log File         D:\oracle\product\10.1.0\db_1\network\log\listener.log

Listening Endpoints Summary...
  (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(PIPENAME=\\.\pipe\EXTPROCipc)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=nbs)(PORT=1521)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=nbs)(PORT=8080))(Presentation=HTTP)(
Session=RAW))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=nbs)(PORT=2100))(Presentation=FTP)(S
ession=RAW))
Services Summary...
Service "OBAYXDB" has 1 instance(s).
  Instance "obay", status READY, has 1 handler(s) for this service...
Service "ORCL" has 1 instance(s).
  Instance "ORCL", status UNKNOWN, has 1 handler(s) for this service...
Service "PLSExtProc" has 1 instance(s).
  Instance "PLSExtProc", status UNKNOWN, has 1 handler(s) for this service...
Service "obay" has 1 instance(s).
  Instance "obay", status READY, has 1 handler(s) for this service...
The command completed successfully
LSNRCTL> _
```

من الإصدار Oracle8i فصاعداً يتم تسجيل الInstance الجديدة الياً فى الDefault Listener اى لا يحتاج مدير قاعدة البيانات إضافتها يدوياً فى ال Listener ، وذلك فيما يعرفDynamic Service registration .
هكذا نكون إنتهينا من تهيئة الجانب الاول وهو المخدم DATABASE SERVER ، وفى الصفحات التالية سنتحدث عن الجانب الاخر وهو جانب الClients.

ليستطيع الClient الاتصال بقاعدة البيانات فى Database Server يحتاج لمعرفة بعض المعلومات الضرورية لنجاح عملية الإتصال :

1- المخدم او الHOST الذى يعمل فيه المستمع Listener.
2- الPort الذى يراقبه المستمع.
3- البروتكول Protocol الذى يستخدمه المستمع.
4- اسم الservice او الInstance الذى يخدمه المستمع.

فعندما يطلب البرنامج او الApplication الإتصال بقاعدة البيانات من خلال المستمع Listener يحتاج لمعرفة المعلومات الموضحة اعلاه ليترجم ذلك فى عملية إتصال ناجح ، أما كيف يعالج هذه المعلومات لتتم عملية الاتصال فهناك عدة طرق :-

1- Easy Connect:-

فى هذا النوع من الإتصال يحتاج المستخدم لكتابة كل المعلومات التى يحتاجها الClient للاتصال بقاعدة البيانات ، يكتب هذه المعلومات اثناء عملية الإتصال وذلك على النحو التالى:-

<username>/<password>@<hostname>:<listener port>/<service name>

```
CONN SYSTEM/SYS@NBS:1521/OBAY
```



حيث NBS: هو اسم الذى يعمل عليه المستمع Listener.
1521: هو الPORT الذى يراقبه المستمع.
OBAY: هو اسم الInstance التى يخدمها المستمع.

وهذا النوع هو اسهل طرق الإتصال من حيث أنه لا يحتاج لتهيئة فى الClient.

2- Local Naming:-

فى النوع السابق من الاتصال Easy Connect تحتاج لكتابة المعلومات وهى (host & protocol & port & service name) اثناء عملية الإتصال مما يسبب نوع المشقة ، فى هذا النوع من الإتصال لا نحتاج الى كتابة ذلك عند كل عملية اتصال وانما نقوم بتهيئة الملف $ORACLE_HOME/network/admin/tnsnames.ora. إذا كنا نعمل على نظام التشغيل UNIX او الملف التالى إذا كنا نعمل على نظام التشغيل WINDOWS. %ORACLE_HOME%\NETWORK\ADMIN\TNSNAMES.ORA

بحيث نقوم بعمل اسم مستعار لكل المعلومات المطلوبة اثناء عملية الإتصال وهى (host & protocol & port & service name) بحيث يمثل هذا الاسم المستعار المعلومات اعلاه ، فلا نحتاج اثناء عملية الاتصال سواء كتابة الاسم المستعار مع اسم المستخدم وكلمة السر username/password@alias ،

نستطيع وضع قائمة من عمليات تهيئة الاسماء المستعارة التى تمثل مجموعة الاتصالات بقاعدة البيانات فى الملف tnsnames.ora.

```
tnsnames.ora - Notepad
File  Edit  Format  View  Help
# tnsnames.ora Network Configuration File: D:\oracle\product\10.1.0\db_1\network\admin\tnsnames.ora
# Generated by Oracle configuration tools.

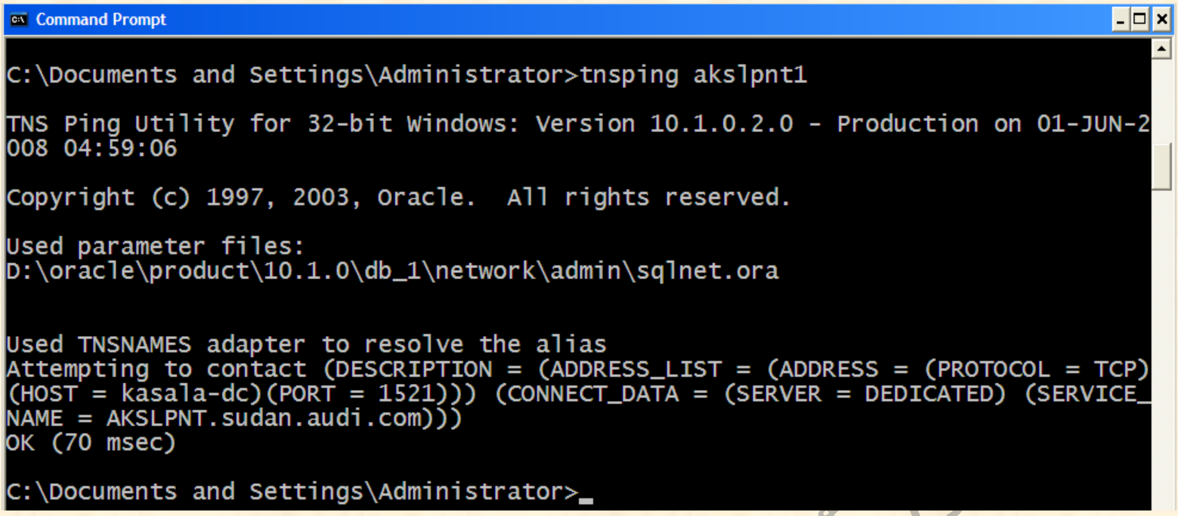

AKSLPNT1 =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = kasala-dc)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = AKSLPNT.sudan.audi.com)
    )
  )
KRT.SUDAN.AUDI.COM =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = 10.122.1.18)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = KKRTPNT.SUDAN.AUDI.COM)
    )
  )

KMTGPNT.SUDAN.AUDI.COM =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = 10.122.8.18)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = KMTGPNT.SUDAN.AUDI.COM)
    )
  )
KMTGPNT =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = 10.122.8.18)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = KMTGPNT.SUDAN.AUDI.COM)
    )
  )
```

هذا نموذج للملف tnsnames.ora . يستطيع هذا الClient الإتصال بقواعد البيانات مستخدماً تهيئة هذا الملف .
ولنفترض أن المستخدم فى هذا الClient كتب العبارة التالية :-
CONN USERNAME/PASSWORD@AKSLPNT1

فإن الClient سيترجم كلمة AKSLPNT1 على النحو التالى :-
HOST: إسم المخدم الذى يحتوى عل المستمع KASALA-DC
PROTOCOL:البرتوكول الذى يستعمله وهو TCP.
PORT:البورت الذى يراقبه المستمع.
SERVICE_NAME:اسم الINSTANCE التى تريد الإتصال بها.

من الممكن القيام بعمل اختبار للOracle Net Service aliases بواسطة الامر tnsping ثم بعده نكتب الاسم المستعار aliases .



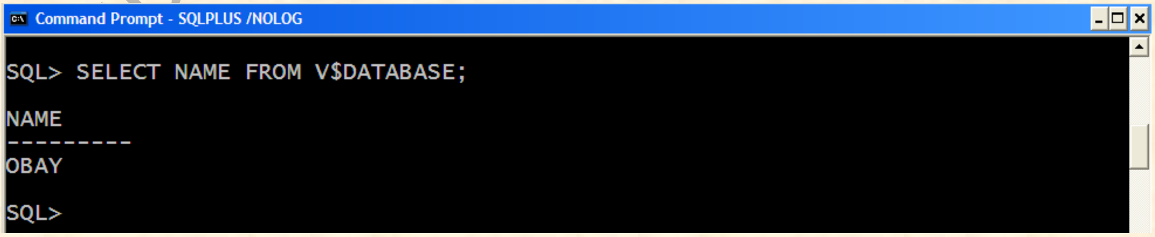
لقد تمت عملية الاختبار بنجاح .

2- Database Link:-

فى قاعدة البيانات الواحدة يستطيع المستخدمون منح صلاحيات على الكائنات التى يملكها لغيره من المستخدمين ، أما إذا كان لديك اكثر من قاعدة بيانات فمن المتعذر تعامل المستخدمين بين قاعدة بيانات واخرى ما لم يكن لدينا Database Link فهى الرابط بين قاعدة بيانات واخرى ، وقد تحتاج ذلك كثيراً اثناء عملك فقد يكون لديك اكثر من قاعدة بيانات تحتاج لربطها مع بعض .

ولنفترض أن لدينا قاعدة بيانات تسمى OBAY

```
SELECT NAME FROM V$DATABASE;
```



واخرى تسمى ORCL.

```
SQL> SELECT NAME FROM V$DATABASE;

NAME
---------
ORCL

SQL>
```

نحتاج الان ربط قاعدة البيانات OBAY مع قاعدة البيانات ORCL ، ولنفترض أن المستخدم TEST فى قاعدة البيانات OBAY يحتاج لعمل استعلام على الجدول EMPLOYEE المملوك للمستخدم VBS الموجود فى قاعدة البيانات ORCL ، إذاً نحتاج لعمل Database Link بين المستخدم TEST فى قاعدة البيانات OBAY وبين المستخدم VBS فى قاعدة البيانات ORCL.

لكن قبل إنشاء الDatabase Link لابد من الإشارة إلى أن المستخدم الذى يقوم بإنشاء الDatabase Link لابد أن يكون لديه الصلاحية CREATE DATABASE LINK.

والان المستخدم TEST فى قاعدة البيانات OBAY سيقوم بعمل Database Link بينه وبين المستخدم VBS فى قاعدة البيانات ORCL.

اولاً: يتصل المستخدم TEST ويتأكد أن لديه الصلاحية CREATE DATABASE LINK.

```
SELECT * FROM USER_SYS_PRIVS WHERE
PRIVILEGE='CREATE DATABASE LINK';
```

```
SQL> CONN /AS SYSDBA
Connected.
SQL> SELECT NAME FROM V$DATABASE;

NAME
---------
OBAY

SQL> CONN TEST/TEST
Connected.
SQL> SELECT * FROM USER_SYS_PRIVS WHERE PRIVILEGE='CREATE DATABASE LINK';

USERNAME                       PRIVILEGE                                ADM
------------------------------ ---------------------------------------- ---
TEST                           CREATE DATABASE LINK                     NO

SQL>
```

في الخطوة السابقة تأكدنا اولاً أننا نعمل على قاعدة البيانات OBAY
وتأكدنا ثانياً أن المستخدم TEST يملك الصلاحية CREATE DATABASE LINK.

ثانياً: يقوم المستخدم TEST بإختبار الOracle Net Service aliases الموجودة فى ملف الtnsnames.ora وذلك للتأكد من سلامة الإتصال بقاعدة البيانات الاخرى وهى هنا ORCL .

```
C:\Documents and Settings\Administrator>TNSPING ORCL

TNS Ping Utility for 32-bit Windows: Version 10.1.0.2.0 - Production on 03-JUN-2
008 05:02:56

Copyright (c) 1997, 2003, Oracle.  All rights reserved.

Used parameter files:
D:\oracle\product\10.1.0\db_1\network\admin\sqlnet.ora


Used TNSNAMES adapter to resolve the alias
Attempting to contact (DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)
(HOST = nbs)(PORT = 1521))) (CONNECT_DATA = (SERVICE_NAME = orcl)))
OK (50 msec)

C:\Documents and Settings\Administrator>_
```

لقد تمت عملية الإختبار بنجاح.

ثالثاً يقوم المستخدم TEST بإنشاء الDatabase Link.

```
CREATE DATABASE LINK TESTVBS CONNECT TO VBS
IDENTIFIED BY VBS USING 'ORCL';
```

لقد قام المستخدم TEST بإنشاء Database Link تسمى TESTVBS يستطيع

```
SQL> SHOW USER
USER is "TEST"
SQL> CREATE DATABASE LINK TESTVBS CONNECT TO VBS IDENTIFIED BY VBS USING 'ORCL';


Database link created.

SQL>
```

من خلالها المستخدم TEST الاتصال بالمستخدم VBS الموجود فى قاعدة بيانات اخرى تسمى ORCL.

رابعاً الان يمكن المستخدم TEST الإستعلام عن الجدول EMPLOYEE المملوك للمستخدم VBS الموجود فى قاعدة البيانات ORCL.

```
SELECT * FROM EMPLOYEE@TESTVBS;
```

```
SQL> SHOW USER
USER is "TEST"
SQL> SELECT * FROM EMPLOYEE@TESTVBS;

    EMP_NO EMP_NAME
---------- -----------------------------------
         1 MOHAMMED

SQL>
```

فى بعد الاحيان يكون اسم الDatabase Link مصحوباً بإسم الجدول طويل فمن الافضل إنشاء مرادف synonym لتسهيل واختصار كتابة العبارات خصوصاً تلك التى يتكرر كتابتها.

```
CREATE SYNONYM EMP FOR EMPLOYEE@TESTVBS;
```

لكن يجب مراعاة أن المستخدم يحتاج للصلاحية CREATE SYNONYM لإنشاء المرادف.

```
Command Prompt - SQLPLUS /NOLOG

SQL> SHOW USER
USER is "TEST"
SQL> CREATE SYNONYM EMP FOR EMPLOYEE@TESTVBS;

Synonym created.

SQL>
```

الان يمكن كتابة عبارة الإستعلام بالشكل الاتى:

```
SELECT * FROM EMP;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> SHOW USER
USER is "TEST"
SQL> SELECT * FROM EMP;

    EMP_NO EMP_NAME
---------- -----------------------------------
         1 MOHAMMED

SQL>
```

تجدر الإشارة هنا بأن نقول انك تستطيع من خلال الDatabase Link إجراء عمليات الDML وهى (SELECT & INSERT & UPDATE &DELETE) لكن من المتعذر اجراء عمليات الDDL مثل CREATE & ALTER & DROP.

```
Command Prompt - SQLPLUS /NOLOG

SQL> DROP TABLE EMPLOYEE@TESTVBS;
DROP TABLE EMPLOYEE@TESTVBS
                   *
ERROR at line 1:
ORA-02021: DDL operations are not allowed on a remote database


SQL>
```

PUBLIC DATABASE LINK:

ما قمنا به فى الخطوات السابقة هو ربط المستخدم TEST فى قاعدة البيانات OBAY مع المستخدم VBS فى قاعدة البيانات ORCL عن طريق DATABASE LINK تسمى TESTVBS.

لكن هل يستطيع اى مستخدم اخر فى قاعدة البيانات OBAY استخدام TESTVBS DATABASE LINK ؟ الجواب وبكل سهولة لا ، وذلك لأن ال Database Link التى قمنا بخلقها هى خاصة بالمستخدم TEST وليست لكل المستخدمين وهى تسمى PRIVATE DATABASE LINK.

ولكى يستطيع كل المستخدمون فى قاعدة البيانات OBAY الإتصال بالمستخدم VBS فى قاعدة البيانات ORCL نحتاج لإنشاء PUBLIC DATABASE LINK.

بالطبع لإنشاء هذا النوع نحتاج للصلاحية CREATE PUBLIC DATABASE LINK.

```
SQL> CONN /AS SYSDBA
Connected.
SQL> SELECT NAME FROM V$DATABASE;

NAME
---------
OBAY

SQL> GRANT CREATE PUBLIC DATABASE LINK TO TEST;

Grant succeeded.

SQL> CONN TEST/TEST
Connected.
SQL> CREATE PUBLIC DATABASE LINK PUBTESTVBS CONNECT TO VBS IDENTIFIED BY VBS USI
NG 'ORCL';

Database link created.

SQL>
```

قام مدير قاعدة البيانات بمنح المستخدم TEST الصلاحية CREATE PUBLIC DATABASE LINK ومن ثم قام المستخدم TEST بإنشاء PUBLIC DATABASE LINK تسمى PUBTESTVBS ، الان يستطيع جميع المستخدمين فى قاعدة البيانات OBAY استخدم PUBTESTVBS DATABASE LINK.

```
SQL> CONN TEST1/TEST1
Connected.
SQL> SELECT * FROM EMPLOYEE@PUBTESTVBS;

    EMP_NO EMP_NAME
---------- -----------------------------------
         1 MOHAMMED

SQL>
```

للإستعلام عن الDATABASE LINK فى قاعدة البيانات
DBA_DB_LINKS
USER_DB_LINKS
ALL_DB_LINKS
V$DBLINK

```
SELECT DB_LINK FROM DBA_DB_LINKS;
```

```
SQL> CONN /AS SYSDBA
Connected.
SQL> SELECT NAME FROM V$DATABASE;

NAME
---------
OBAY

SQL> SELECT DB_LINK FROM DBA_DB_LINKS;

DB_LINK
--------------------------------------------------------------------------------

PUBTESTVBS
TESTVBS

SQL>
```

بالطبع يمكن حذف الDATABASE LINK.

```
DROP DATABASE LINK TESTVBS;
```

```
SQL> CONN TEST/TEST
Connected.
SQL> DROP DATABASE LINK TESTVBS;

Database link dropped.

SQL>
```

كما يمكن حذف ال PUBLIC DATABASE LINK

```
DROP PUBLIC DATABASE LINK PUBTESTVBS;
```

```
SQL> CONN /AS SYSDBA
Connected.
SQL> GRANT DROP PUBLIC DATABASE LINK TO TEST;

Grant succeeded.

SQL> CONN TEST/TEST
Connected.
SQL> DROP PUBLIC DATABASE LINK PUBTESTVBS;

Database link dropped.

SQL>
```

بالطبع نحتاج للصلاحية DROP PUBLIC DATABASE LINK للحذف.

3- Materialized Views:-

عند مناقشتنا للDatabase Link عرفنا كيف يمكن ربط قاعدة بيانات بأخرى وكيف يمكن الوصول للكائنات فى قاعدة بيانات من قاعدة بيانات اخرى ، ولكن قد تحتاج فى بعض الاحيان لنقل وتحديث البيانات الموجودة فى قاعدة البيانات الى قاعدة بيانات أخرى .

كأن تكون مثلاً مديراً لمجموعة صيدليات ؛ كل صيدلية تحتوى على قاعدة بيانات ولكن على رأس كل ساعة تحتاج لجلب جميع حسابات الصيدليات من قواعد البيانات الموزعة الى قاعدة البيانات الرئيسية ، هذه هى مهمة الMaterialized Views ؛ ولكن للMaterialized Views مهام اخرى ليست هنا مجال تفصيلها.

سنستخدم هنا نفس المثال الذى استخدمناه فى الDatabase Link ، ولنفترض هنا أن قاعدة البيانات OBAY هى قاعدة البيانات الرئيسية التى ستستقبل البيانات ، ولنفترض أننا سنستقبل البيانات فى المستخدم TEST ، أما قاعدة البيانات الاخرى ORCL والتى تحتوى على الجدول الرئيسى ولنفترض أنه EMPLOYEE المملوك للمستخدم VBS الذى نحتاج الى نقل وتحديث بياناته كل ثانية الى قاعدة البيانات الرئيسية OBAY.

بالطبع نحتاج قبل كل شئ لعمل Database Link بين المستخدم TEST فى قاعدة البيانات OBAY وبين المستخدم VBS فى قاعدة البيانات ORCL ،

وذلك لعملية نقل وتحديث بيانات الجدول EMPLOYEE من قاعدة البيانات ORCL إلى قاعدة البيانات OBAY.

أيضاً عملية تحديث البيانات (Refresh) بين الجدول الرئيسى وال Materialized Views تنقسم الى ثلاث أنواع :

1- REFRESH FAST:- فى هذا النوع من التحديث يتم فقط نقل البيانات التى تغيرت بعد اخر تحديث فهو لا يحتاج لنقل جميع البيانات ف الجدول وانما فقط ما تم تغيره بعد اخر تحديث ، هذا النوع بفى الغالب يختصر الزمن .

2- REFRESH COMPLETE: أما فى هذا النوع من التحديث يقوم بنقل جميع بيانات الجدول من المصدر الى الMaterialized Views فيقوم بعمل إعادة كتابة البيانات القديمة واضافة البيانات الجديدة. ف الغالب ان هذا النوع يتطلب زمن اكثر من النوع FAST.

3- FORCE REFRESH: هذا النوع يبدأ اول بتطبيق النوع FAST إذا فشلت العملية كأن لم يجد مثلاً الMaterialized Views Logs فى جانب المصدر. ففى هذه الحالة يطبق النوع COMPLETE.

إذا لم يحدد نوع التحديث اثناء إنشاء الMaterialized Views فان الاصل DEFAULT هو FORCE.

Materialized Views Logs:-

كما سبق وان ذكرنا أن نوع التحديث FAST يقوم فقط بنقل البيانات التى تم تغيرها منذ اخر تحديث من الجدول المصدر الى الMaterialized Views.
ولكن أين يتم تخزين معلومات عن البيانات التى تغيرت فى الجدول المصدر قبل نقلها الى الMataerialized Views ؟ الجواب يتم تخزينها فى ال Materialized View Logs ، وهو عبارة عن جدول يتم إنشاؤه فى قاعدة البيانات بل وفى المستخدم الذى يملك الجدول المصدر وذلك عن طريق الامر التالى :

```
CREATE MATERIALIZED VIEW LOG ON EMPLOYEE;
```

بحيث EMPLOYEE هو الجدول المصدر.

فلحظة كتابة الامر اعلاه ؛ قاعدة البيانات تقوم بإنشاء جدول بالصيغة <MLOG$_<TABLE_NAME

:Primary Key Materialized Views

ولكن حتى تستطيع تنظيم الجدول المصدر ، دون التأثير على الFAST REFRESH وذلك بحيث يتم حفظ التغيرات بناءً على الPrimary Key ، فنستطيع لحظة إنشاء الMaterialized Views تحديد الخيار WITH PRIMARY KEY وهو الاصل DEFAULT فى حالة عدم تحديد خيار اخر ، والخيار الاخر هو ROWID.

لحظة إنشاء الMaterialized Views بالخيار Primary Key يجب أن يكون الجدول المصدر يحتوى على prmary Key Constraint.

كذلك عند تحديد الخيار Fast Refresh عند إنشاء الMaterialized Views يجب أن نكون قمنا بإنشاء الMaterialized View Logs فى المستخدم الذى يحوى الجدول المصدر وإلا ظهرت لنا رسالة خطأ.

قد يكون فى الشرح السابق نوع من التعقيد لكن لا عليك ركز معى فى السناريو الذى سننفذه معاً خطوة خطوة ابتداءً من إنشاء المستخدمين وحتى عمل الMaterialized Views وهذا هو بيت القصيد.

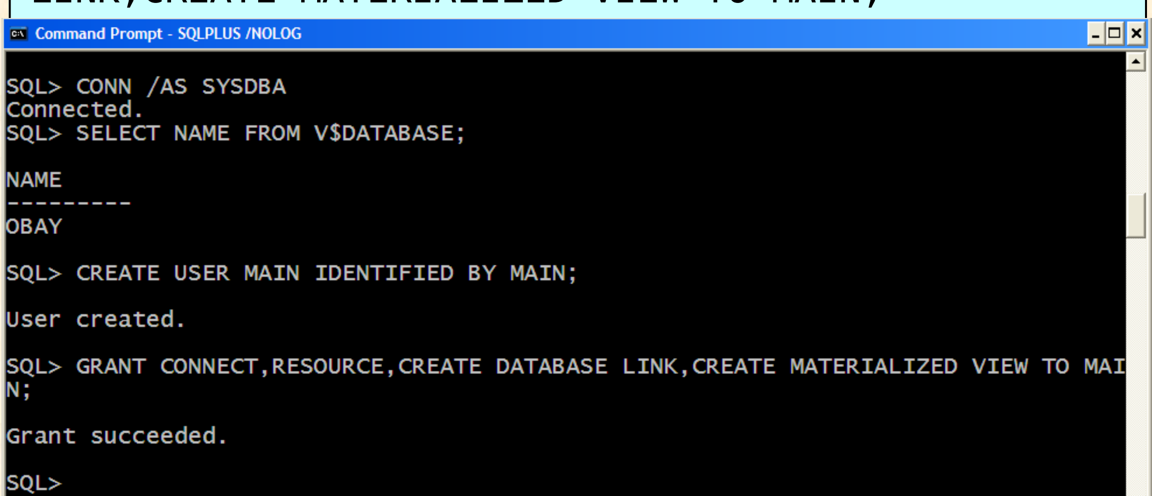
قاعدة البيانات الاولى تسمى OBAY ، سنقوم بإنشاء مستخدم فيها يسمى MAIN ؛ وسنقوم بإنشاء Materialized Views فى هذا المستخدم لجلب بيانات موجودة فى الجدول EMPLOYEE المملوك للمستخدم SUB الموجود فى قاعدة البيانات ORCL.

واليك الخطوات :-

1- سنقوم بإنشاء المستخدم MAIN فى قاعدة البيانات OBAY وسنمنحه الصلاحيات الكافية.

```
CREATE USER MAIN IDETIFIED BY MAIN;
```

```
GRANT CONNECT,RESOURCE,CREATE DATABASE
LINK,CREATE MATERIALIZED VIEW TO MAIN;
```



2- فى الجانب الاخر نتأكد من الجدول المصدر وهو هنا EMPLOYEE والتأكد كذلك أنه يحتوى على PRIMARY KEY CONSTRAINT حتى نستطيع إنشاء MATERIALIZED VIEWS فى قاعدة البيانات الاخرى بإستخدام الخيار WITH PRIMARY KEY.

```
Command Prompt - SQLPLUS /NOLOG

SQL> CONN /AS SYSDBA
Connected.
SQL> SELECT NAME FROM V$DATABASE;

NAME
---------
ORCL

SQL> CONN SUB/SUB
Connected.
SQL> SELECT * FROM EMPLOYEE;

    EMP_NO EMP_NAME
---------- -----------------------------------
         1 MOHAMMED
         2 AHMED

SQL> SELECT OWNER,CONSTRAINT_NAME,CONSTRAINT_TYPE,TABLE_NAME FROM USER_CONSTRAIN
TS;

OWNER                          CONSTRAINT_NAME                C
------------------------------ ------------------------------ -
TABLE_NAME
------------------------------
SUB                            PK_EMP_NO                      P
EMPLOYEE


SQL>
```

```
SELECT * FROM EMPLOYEE;
```

إذاً الجدول EMPLYEE يحتوى على حقلين ويحتوى كذلك على CONSTRAINT

```
SELECT
OWNER,CONSTRAINT_NAME,CONSTRAINT_TYPE,TABLE_NAME
FROM USER_CONSTRAINTS;
```

PRIMARY KEY.

3- الان فى قاعدة البيانات OBAY نقوم بإنشاء DATABASE LINK بين المستخدم MAIN وبين المستخدم SUB فى قاعدة البيانات ORCL.

```
CREATE DATABASE LINK MAINSUB CONNECT TO SUB
IDENTIFIED BY SUB USING 'ORCL';
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> CONN /AS SYSDBA
Connected.
SQL> SELECT NAME FROM V$DATABASE;

NAME
---------
OBAY

SQL> CONN MAIN/MAIN
Connected.
SQL> CREATE DATABASE LINK MAINSUB CONNECT TO SUB IDENTIFIED BY SUB USING 'ORCL';


Database link created.
```

4- فى قاعدة البيانات ORCL وفى المستخدم SUB الذى يحوى الجدول المصدر EMPLOYEE نقوم بإنشاء MATERIALIZED VIEW LOG.

```
Command Prompt - SQLPLUS /NOLOG

SQL> CONN /AS SYSDBA
Connected.
SQL> SELECT NAME FROM V$DATABASE;

NAME
---------
ORCL

SQL> CONN SUB/SUB
Connected.
SQL> CREATE MATERIALIZED VIEW LOG ON EMPLOYEE;

Materialized view log created.
```

قمنا بإنشاء MATERIALIZED VIEW LOG للجدول EMPLOYEE ويمكن التحقق من إنشاء الMATERIALIZED VIEW LOG للجدول EMPLOYEE بالإستعلام التالى

```
SELECT * FROM TAB WHERE TNAME LIKE '%EMPLOYEE';
```

5- فى قاعدة البيانات OBAY نقوم بإنشاء الMATERIALIZED VIEW.

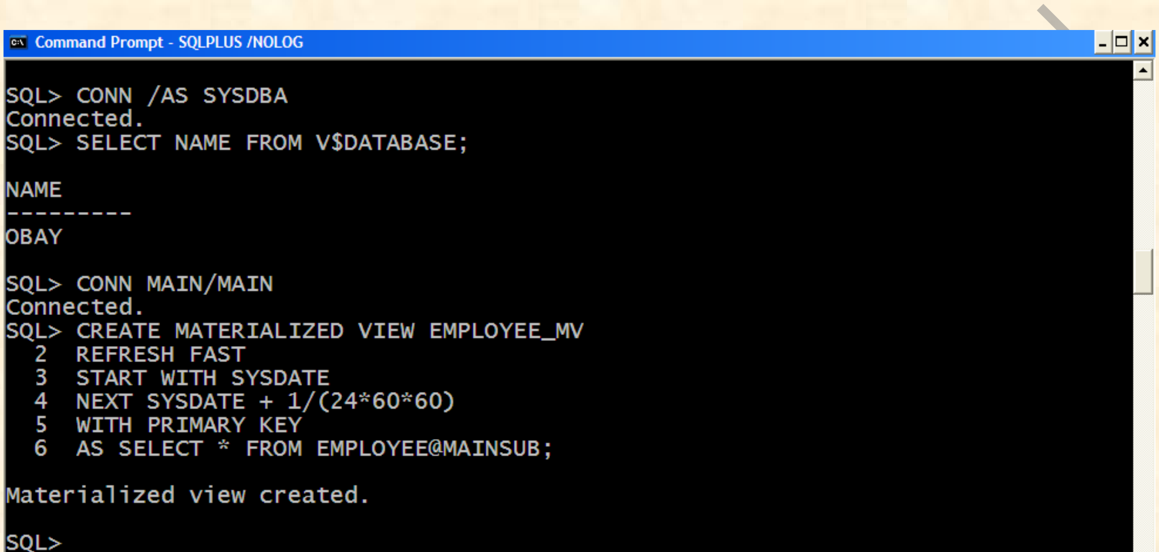
```
Command Prompt - SQLPLUS /NOLOG

SQL> SELECT * FROM TAB WHERE TNAME LIKE '%EMPLOYEE';

TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
EMPLOYEE                       TABLE
MLOG$_EMPLOYEE                 TABLE
RUPD$_EMPLOYEE                 TABLE

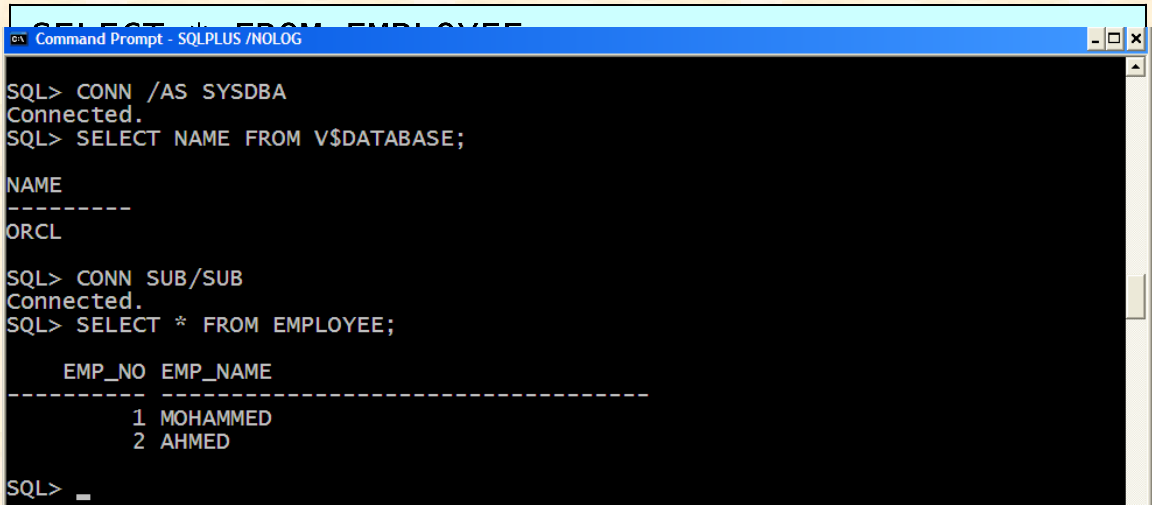
SQL> _
```

```
CREATE MATERIALIZED VIEW EMPLOYEE_MV
REFRESH FAST
START WITH SYSDATE
NEXT SYSDATE + 1/(24*60*60)
WITH PRIMARY KEY
AS SELECT * FROM EMPLOYEE@MAINSUB;
```



قمنا بإنشاء الMATERIALIZED VIEW مستخدمين الخيار FAST REFRESH وذلك لأننا قمنا بإنشاء الMATERIALIZED VIEW LOG فى الجانب الاخر وإلا ظهرت رسالة خطأ ، وكذلك استخدمنا الخيار WITH PRIMARY KEY وذلك لأن الجدول المصدر يحتوى على PRIMARY KEY CONSTRAINT ، على اى حال هناك خيار اخر هو WITH ROWID.

6- الان يمكن الإستعلام عن الجدول المصدر والMATERIALIZED VIEW ستلاحظ أنه كل ثانية كما حددنا ذلك [SYSDATE +1/(24*60*60]تكون النتيجة متطابقة بين الجدول المصدر EMPLOYEE وبين الMATERIALIZED VIEW وهى EMPLOYEE_MV.



```
SELECT * FROM EMPLOYEE_MV;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> CONN /AS SYSDBA
Connected.
SQL> SELECT NAME FROM V$DATABASE;

NAME
---------
OBAY

SQL> CONN MAIN/MAIN
Connected.
SQL> SELECT * FROM EMPLOYEE_MV;

    EMP_NO EMP_NAME
---------- -----------------------------------
         1 MOHAMMED
         2 AHMED

SQL>
```

7- الان لو قمنا بإضافة حقل جديد فى الجدول المصدر ثم بعد ذلك أعدنا عمليات الإستعلام اعلاه.

```
INSER INTO EMPLOYEE VALUES (3,'ALI');
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> CONN /AS SYSDBA
Connected.
SQL> SELECT NAME FROM V$DATABASE;

NAME
---------
ORCL

SQL> CONN SUB/SUB
Connected.
SQL> INSERT INTO EMPLOYEE VALUES(3,'ALI');

1 row created.

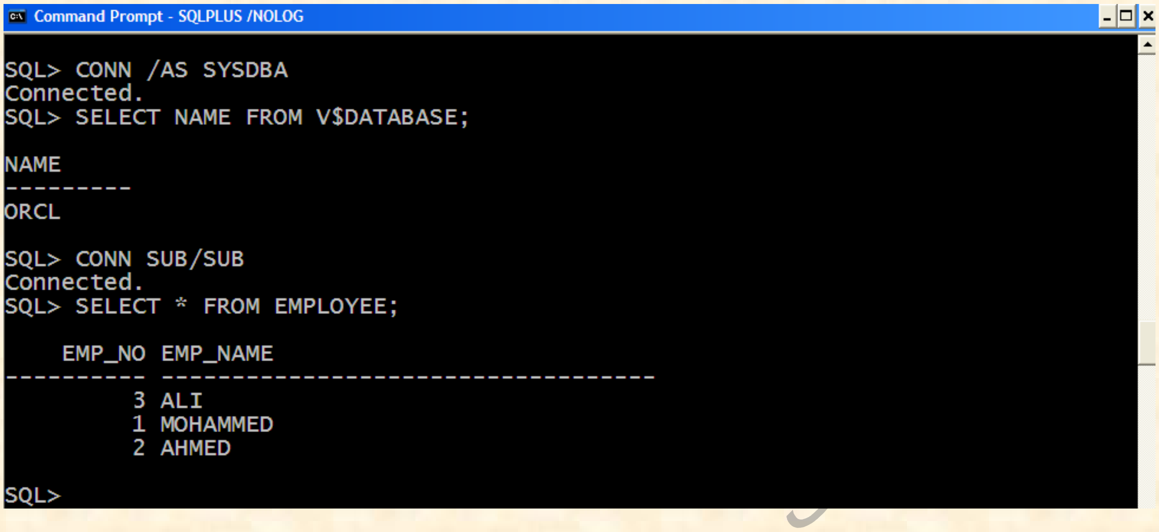
SQL> COMMIT;

Commit complete.

SQL>
```

لاحظت معى إضافة حقل ثالث فى الجدول EMPLOYEE.

ستنعكس الإضافة فى الجدول المصدر EMPLOYEE على EMPLOYEE_MV
MATERIALIZED VIEW بعد ثانية واحدة من الاضافة فى الجدول المصدر.



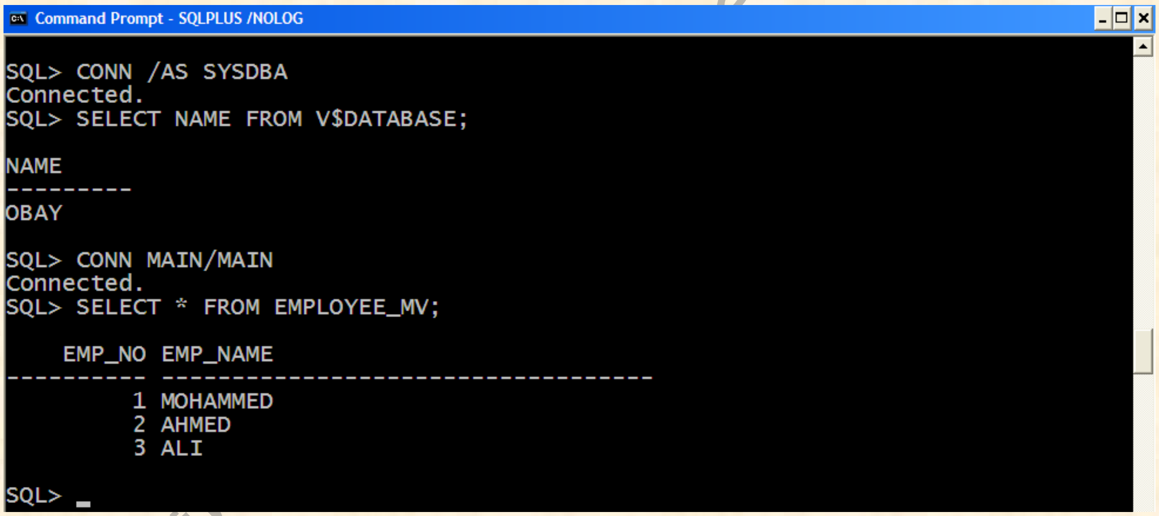
```
SQL> CONN /AS SYSDBA
Connected.
SQL> SELECT NAME FROM V$DATABASE;

NAME
---------
ORCL

SQL> CONN SUB/SUB
Connected.
SQL> SELECT * FROM EMPLOYEE;

    EMP_NO EMP_NAME
---------- -----------------------------------
         3 ALI
         1 MOHAMMED
         2 AHMED

SQL>
```



```
SQL> CONN /AS SYSDBA
Connected.
SQL> SELECT NAME FROM V$DATABASE;

NAME
---------
OBAY

SQL> CONN MAIN/MAIN
Connected.
SQL> SELECT * FROM EMPLOYEE_MV;

    EMP_NO EMP_NAME
---------- -----------------------------------
         1 MOHAMMED
         2 AHMED
         3 ALI

SQL> _
```

لو قمنا بعمل إستعلام للجدول MLOG$_EMPLOYEE قبل عملية ال REFRESH سنجد معلومات عن الحقول التى اضيفت بعد اخر عملية REFRESH ولكن بعد عملية الREFRESH تختفى المعلومات السابقة عن الجدول وفى إنتظار الجديد.

يستطيع مدير قاعدة البيانات عمل إستعلام عن الMATERIALIZED VIEWS التى تعمل فى قاعدة البيانات بواسطة الجدول
DBA_MVIEWS

```
Command Prompt - SQLPLUS /NOLOG

SQL> CONN /AS SYSDBA
Connected.
SQL> SELECT NAME FROM V$DATABASE;

NAME
---------
OBAY

SQL> SELECT OWNER,MVIEW_NAME,REFRESH_MODE,QUERY FROM DBA_MVIEWS;

OWNER                          MVIEW_NAME                     REFRES
------------------------------ ------------------------------ ------
QUERY
--------------------------------------------------------------------------------

MAIN                           EMPLOYEE_MV                    DEMAND
SELECT "EMPLOYEE"."EMP_NO" "EMP_NO","EMPLOYEE"."EMP_NAME" "EMP_NAME" FROM "EMPLO


SQL>
```

نستطيع أن نحصل على معلومات اخرى من نفس الجدول.
للاستعلام عن الJOBS بواسطة الجدول DBA_JOBS.

```
Command Prompt - SQLPLUS /NOLOG

SQL> CONN /AS SYSDBA
Connected.
SQL> SELECT NAME FROM V$DATABASE;

NAME
---------
OBAY

SQL> SELECT JOB,SCHEMA_USER,LAST_DATE,LAST_SEC,NEXT_SEC FROM DBA_JOBS WHERE LOG_
USER='MAIN';

       JOB SCHEMA_USER                    LAST_DATE LAST_SEC NEXT_SEC
---------- ------------------------------ --------- -------- --------
        27 MAIN                           06-JUN-08 14:28:23 14:28:24

SQL>
```

بالطبع المستخدم MAIN يستطيع حذف الMATERIALIZED VIEW.

```
DROP MATERIALIZED VIEW EMPLOYEE_MV;
```

```
Command Prompt - SQLPLUS /NOLOG
SQL> CONN MAIN/MAIN
Connected.
SQL> DROP MATERIALIZED VIEW EMPLOYEE_MV;

Materialized view dropped.

SQL>
```

الفصل التاسع

:Undo Management & Flashback Technology

:UNDO MANAGEMENT -1

فى الفصل الثالث من هذا الكتاب تحدثت عرضاً عن الUndo Tablespace وذكرت أنه يستخدم من قبل الOracle Server لتخزين ال Undo Information ، لكن هل فكرت يوماً فى عملية الROLLBACK ؟ وهل سألت نفسك كيف يمكنك التراجع عن عمليات التعديل التى قمت بها فى بعض الحقول؟ ، رغم أنه تم تغير القيم القديمة الى القيم الجديدة .

السناريو بإختصار أنه عند إجراء عملية تعديلات على البيانات فى قاعدة البيانات فإن الOracle Server يقوم يتخزين القيم القديمة فى الUndo Tablespace قبل ان يتم تغييرها بالقيم الجديدة ، هذا السناريو يتيح لنا فرصة التراجع عن العمليات متى احتجنا إلى ذلك بشرط أن تكون القيم القديمة ما زالت موجودة فى الUndo Tablespace إذ لا يتصور أن تظل هذه القيم موجودة إلى مالانهاية فهو فى اخر الامر Tablespace له سعة محدودة وكذلك له فترة احتفاظ

بالمعلومات محدودة يقوم بتهيئتها مدير قاعدة البيانات حسب ما يراه مناسباً ، نستطيع كذلك من خلال هذا السناريو ان ننجز عملية الFlashback Queries إذا توفرت القيم القديمة سنتحدث عن الFlashback Queries فى هذا الفصل ولكن عموماً هو عبارة عن استعلام عن قيم لكن فى الماضى عند زمن معين .

فى الاصدارات السابقة كان يتم تخزين القيم التى تم تعديلها فى قاعدة البيانات فى Rollback Segment ولكن بعد الإصدار Oracle9i فصاعداً قدمت شركة اوركل خيار جديد هو Undo Segment مع إبقاء الخيار الاول متاحاً ، ولكن تنصح شركة اوركل بقوة استخدام الUndo Segment.

فالUndo Tablespace تتم إدارته عن طريق Managed Tablespace Locally وايضاً Automatic Extent allocation ، فلحظة بدء العملية يتم تخصيص وإنشاء Undo Segment الياً لتخزين القيم التى تتغير فى قاعدة البيانات بواسطة هذه العملية ، بحيث يتم تخصيص Undo Segment واحدة لكل عمليات ولكن يمكن أن تخدم هذه الSegment عدد من العمليات، فعند امتلاء ال Extent يتم التحول للتى بعدها فى نفس الSegment بحيث يكون على الاقل هناك اثنين من الExtents فى كل Segement أما الحد الاقصى فيعتمد على DB Block Size ، أما إذا تم ملء جميع الExtents فى الUndo Segment فإنه يتم إعادة الكتابة فى الExtent إبتداءً من الأول ، أو يتم طلب تخصيص Extent جديدة.

قد يكون فى قاعدة البيانات الواحدة اكثر من Undo Tablespace لكن لا يمكن ان يعمل فى اللحظة الواحدة اكثر من Undo Tablespace ويتم تحديد عمل الUndo Tablespace فى قاعدة البيانات بواسطة المتغير Undo_Tablespace.

```
Command Prompt - SQLPLUS /NOLOG

SQL> SHOW PARAMETER UNDO_TABLESPACE;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
undo_tablespace                      string      UNDOTBS1
SQL>
```

لإنشاء Undo Tablespace جديد

```
CREATE UNDO TABLESPACE ORCLUNDO DATAFILE 'D:\
oracle\product\10.1.0\oradata\ORCL\ORCLUNDO.DBF'
SIZE 20M EXTENT MANAGEMENT LOCAL AUTOALLOCATE;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> CREATE UNDO TABLESPACE ORCLUNDO DATAFILE 'D:\oracle\product\10.1.0\oradata\
ORCL\ORCLUNDO.DBF' SIZE 20M EXTENT MANAGEMENT LOCAL AUTOALLOCATE;

Tablespace created.

SQL>
```

الان قاعدة البيانات تحتوى على اثنين من الUndo Tablespace ولكن يعمل واحد فقط هو UNDOTBS1 كما موضح اعلاه .

ولكن يمكن تغيير الUndo Tablespace الذى يعمل فى قاعدة البيانات من UNDOTBS1 الى ORCLUNDO.

```
ALTER SYSTEM SET UNDO_TABLESPACE='ORCLUNDO'
SCOPE=BOTH;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> ALTER SYSTEM SET UNDO_TABLESPACE='ORCLUNDO' SCOPE=BOTH;

System altered.

SQL> SHOW PARAMETER UNDO_TABLESPACE;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
undo_tablespace                      string      ORCLUNDO
SQL>
```

قاعدة البيانات اوركل Oracle Database10g تتيح خيار ادرة الياً لل Undo Tablespace وهو Automatic Undo Management ويتم تهيئته بواسطة المتغير UNDO_MANAGEMENT بحيث يأخذ القيمة AUTO ، أما خيار الإدارة اليدوى Manual Undo Magamenet فهو خيار مكلف ويحتاج لعمل اكثر من مدير قاعدة البيانات .

أما خيار الإدارة Automatic Undo Management فهو يقلل العبء من مدير قاعدة البيانات بحيث تكون إدراته فقط على مستوى الTablespace.

مدير قاعدة البيانات ينتظره تهيئة المساحة المناسبة للUndo Tablespace وذلك حسب المعلومات التى سيتم تخزينها ، كذلك لابد من تهيئة فترة الإحتفاظ Undo Retention للبيانات فى الUndo Segment وذلك بواسطة المتغير UNDO_RETENTION الذى يأخذ قيمة هى فترة الاحتفاظ بالثوانى .

الاصل فى هذا المتغير هو أن يأخذ القيمة 0 وهى تعنى Automatic اى يعنى محاولة الاحتفاظ بالمعومات حتى تنتهى ، على أن يتم الإحتفاظ بالمعلومات على الاقل 15 دقيقة ، لكن يمكن تحديد قيمة اخرى هى فترة الاحتفاظ بالثوانى واقصى قيمة هى 2^{32}.

```
SHOW PARAMETER UNDO_RETENTION;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> SHOW PARAMETER UNDO_RETENTION;

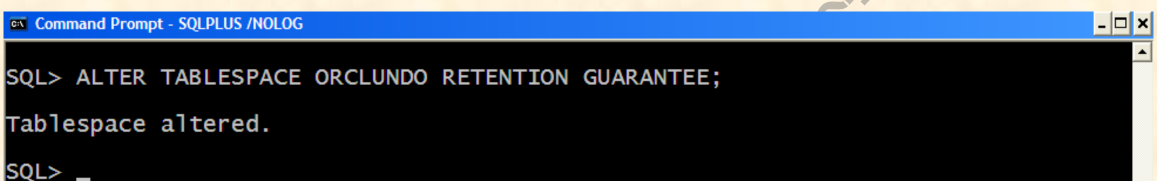
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
undo_retention                       integer     900
SQL>
```

وعموماً Undo information اى المعلومات الموجودة فى الUndo Segments تنقسم الى ثلاث حالات :-

1- Uncommitted Undo Information: وهى المعلومات التى لم يتم تثبيتها إلى الان وذلك لأن العمليات مازالت مستمرة ، هذا النوع من المعلومات لا يمكن حذفها واعادة الكتابة فيها .

2- Committed Undo Information: وهى لا نحتاجها لعمليات مستمرة ، ولكن لأن فترة الإحفاظ لم تنتهى بعد "Unexpired"، هذا النوع من العمليات نحتفظ به قدر الإمكان ما لم يؤدى ذلك لفشل بعض العمليات نتيجة عدم وجود مساحة فى الUndo Tablespace فى هذه الحالة يتم اعادة الكتابة فى هذه المعلومات ، ولكن قد يقوم مدير قاعدة البيانات بتهيئة الUndo Tablespace بحيث نضمن عدم مسح واعادة الكتابة فيها وذلك باستخدام الخيار Guaranteeing Undo Retention.

```
ALTER TABLESPACE ORCLUNDO RETENTION GUARANTEE;
```



هذا الخيار بالطبع غير متاح سواء للUndo Tablespace.
هكذا نضمن عدم مسح المعلومات التى لم تنتهى فترة احتفاظها حتى لو أدى ذلك لفشل بعض العمليات لعدم وجود مساحة فى الUndo Tablespace.

3- Expired Undo Information: وهى لا نحتاجها لعمليات مستمرة ، وكذلك فترة الاحتفاظ بها انتهت فيمكن اعادة الكتابة فيها متى ما احتجنا لمساحة فى ال Undo tablespace.

من المشاكل التى تحدث كثيراً والتى يجب لمدير قاعدة البيانات مراعاتها :-

1- مشكلة المساحة Undo Tablespace Space Error: ويجب على مدير قاعدة البيانات مراقبة مساحة الUndo Tablespace فالعمليات التى لا تجد مساحة فى الUndo Tablespace تعطى رسالة الخطأ.
ORA-01650: unable to extend rollback segment

2- “Snapshot too old” Error: وهذا الخطأ يظهر عند تنفيذ استعلام يحتاج لمعلومات Undo Information قد تم مسحها واعادة الكتابة فيها ، لذا يجب على مدير قاعدة البيانات مراعاة فترة الإحتفاظ المناسبة UNDO_RETENTION وكذلك المساحة المناسبة مع مراعاة Guaranteeing Undo Retention.

يمكن الإستعلام عن الUNDO بواسطة:

DBA_UNDO_EXTENT
V$UNDOSTAT

:Flashbach Technology -2

:A- Flashback Query
From Oracle91 & Do use Undo

:B- Flashback Table
From Oracle10g & Do use Undo

:C- Flashback Version Query
From Oracle10g & Do use Undo

:D- Flashback Transaction

From Oracle10g & Do use Undo

:E- Flashback Drop
From Oracle10g & Do Not use Undo

:F- Flashback Database
From Oracle10g & Do Not use Undo

:A- Flashback Query
هذا النوع متاح منذ الإصدار Oracle9i بحيث نستطيع من خلاله عمل إستعلام فى قاعدة البيانات ليس للبيانات الموجودة حالياً وإنما لبيانات موجودة فى لحظة زمنية فى الماضى ، ويعتمد هذا النوع اساساً على الUndo Tablespace ، فبمجرد عمل الاستعلام يتم البحث عن المعلومات فى ال Undo Segments ، هذه المعلومات التى ستعرض للمستخدم تظل مؤقتة ومتاحة فقط للSession الحالية ، بالطبع فقد تفشل عملية الFlashback Query إذا لم نجد البيانات المطلوبة فى الUndo Tablespace مثلاً بسبب طول المدة وإعادة كتابة بيانات جديدة فى البيانات المطلوبة لذا على مدير قاعدة البيانات تحديد فترة احتفاظ Retention مناسبة.

ولنتصور الان السناريو التالى حتى نستوعب عملية الFlashback Query بصورة اوضح:
لدينا الان جدول يسمى Employee يحتوى على على مجموعة من الحقول.

```
SELECT * FROM EMPLOYEE;
```

```
SQL> SELECT * FROM EMPLOYEE;

    EMP_NO EMP_NAME
---------- -----------------------------------
         1 sara
         2 OBAY
         3 SALAH
         4 TOTA
         5 TOTAAA
         6 MAMA
         7 MAMAWW
       100 SOSO

8 rows selected.

SQL>
```

الجدول EMPLOYEE يحتوى الان على 8 حقول.

لنفترض أن المستخدم قام بمسح بعض الحقول .

```
DELETE EMPLOYEE WHERE EMP_NO IN (1,5,7);
```

```
SQL> DELETE EMPLOYEE WHERE EMP_NO IN (1,5,7);

3 rows deleted.

SQL> COMMIT;

Commit complete.

SQL>
```

الان لو قمنا بإستعراض الجدول.

```
SQL> SELECT * FROM EMPLOYEE;

    EMP_NO EMP_NAME
---------- -----------------------------------
         2 OBAY
         3 SALAH
         4 TOTA
         6 MAMA
       100 SOSO

SQL>
```

بقى لدينا فقط 5 حقول .

ماذا لو قمنا الان بعمل استعلام على الجدول ولكن فى فترة زمنية فى الماضى ولنفترض انها قبل 20 دقائق .
بالطبع سيبحث عن هذه المعلومات فى الUndo segements ويقوم بعرضها ،
أما إذا لم يجدها فستظهر رسالة خطأ.
اى هذه المعلومات غير متوفرة حالياً فى الUndo Tablespace.

```
SELECT * FROM EMPLOYEE AS OF TIMESTAMP
TO_TIMESTAMP ('11-06-08:03-33-38','DD-MM-
YY:HH24-MI-SS');
```

```
Command Prompt - sqlplus /nolog

SQL> alter session set nls_date_format='dd-mm-yy:hh24-mi-ss';

Session altered.

SQL> SELECT SYSDATE FROM DUAL;

SYSDATE
-----------------
11-06-08:03-53-33

SQL> SELECT * FROM EMPLOYEE AS OF TIMESTAMP TO_TIMESTAMP('11-06-08:03-33-38','DD
-MM-YY:HH24-MI-SS');

    EMP_NO EMP_NAME
---------- -----------------------------------
         1 sara
         2 OBAY
         3 SALAH
         4 TOTA
         5 TOTAAA
         6 MAMA
         7 MAMAWW
       100 SOSO

8 rows selected.

SQL>
```

هذه هى نتيجة الإستعلام ، عرض 8 حقول ، لكن ما هو متاح حالياً هو 5 حقول فقط.

هكذا قمنا بعرض معلومات فى الماضى ، لكن ماذا لو أردنا أن نأتى بهذه المعلومات فى الحاضر أى تظل هذه المعلومات موجودة فى الجدول ، يمكن أن ننشئ جدول جديد من المعلومات المستعرض ومن ثم نقوم بحذف الجدول Employee وإعادة تسمية الجدول الجديد الى Employee1.

```
CREATE TABLE EMPLOYEE1 AS (SELECT * FROM
EMPLOYEE AS OF TIMESTAMP TO_TIMESTAMP('11-06-
08:03-33-38','DD-MM-YY:HH24-MI-SS'));
```

```
Table created.

SQL>
```

الان نقوم بحذف الجدول Employee.

```
DROP TABLE EMPLOYEE;
```

```
Command Prompt - sqlplus /nolog

SQL> DROP TABLE EMPLOYEE;
```

```
Command Prompt - sqlplus /nolog

SQL> RENAME EMPLOYEE1 TO EMPLOYEE;

Table renamed.

SQL> SELECT * FROM EMPLOYEE;

    EMP_NO EMP_NAME
---------- ----------------------------------
         1 sara
         2 OBAY
         3 SALAH
         4 TOTA
         5 TOTAAA
         6 MAMA
         7 MAMAWW
       100 SOSO

8 rows selected.

SQL>
```

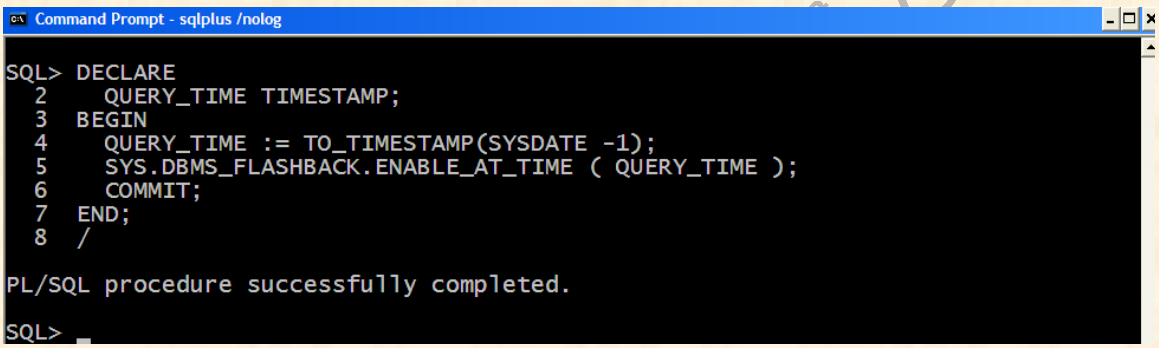
ثم نقوم بإعادة تسمية للجدول EMPLOYEE1 إلى EMPLOYEE.

الان الجدول EMPLOYEE يحتوى على المعلومات التى كانت به منذ 20 دقيقة.

فى الخطوات السابقة قمنا بعمل استعلام فى Session معينة عن بيانات فى لحظة معينة فى الماضى ، ولكن من الممكن كذلك أن نضع كل هذه الِ Session فى لحظة معينة من الماضى ، بحيث تكون نتيجة جميع الإستعلامات لقاعدة البيانات من خلال هذه الِSession فى نقطة معينة من الماضى يتم تحديدها اثناء عمل الFlashback ، ولكن باقى الِSessions التى تعمل على قاعدة البيانات ترى قاعدة البيانات فى الوقت الحقيقى الان ؛ ما عدا هذه ال Session التى ترى قاعدة البيانات فى لحظة معينة من الماضى ، يتم ذلك من خلال الحزمة DBMS_FLASHBACK.

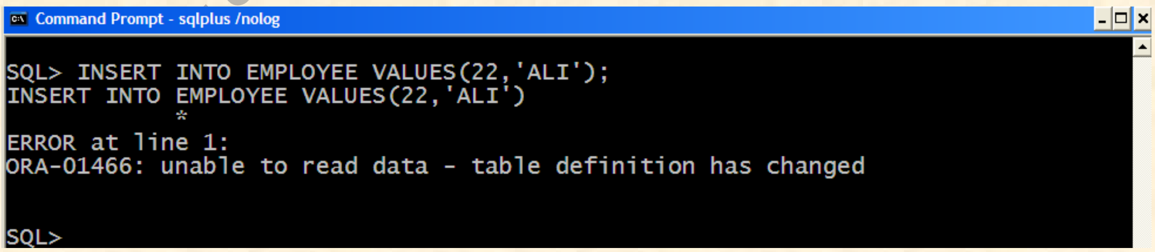
للرجوع بالSession الحالية للماضى يوماً كاملاً نقوم بتنفيذ الإجراء PROCEDURE ENABLE_AT_TIME الموجود فى الحزمة PACKAGE DBMS_FLASHBACK

```
DECLARE
  QUERY_TIME TIMESTAMP;
BEGIN
  QUERY_TIME := TO_TIMESTAMP(SYSDATE -1);
  SYS.DBMS_FLASHBACK.ENABLE_AT_TIME ( QUERY_TIME
);
  COMMIT;
END;
/
```



هكذا خلال هذه الSession جميع الإستعلامات ترى قاعدة البيانات كما لو كنا بالامس.

لابد من الإشارة إلى أنه من خلال هذا النمط لا يمكن إجراء عمليات الDML ماعدا SELECT.



يمكن إلغاء هذا النمط بواسطة الاجراء DISABLE الموجود فى الحزمة DBMS_FLASHBACK

```
Command Prompt - sqlplus /nolog

SQL> BEGIN
  2     SYS.DBMS_FLASHBACK.DISABLE;
  3     COMMIT;
  4
  5   END;
  6   /

PL/SQL procedure successfully completed.

SQL>
```

هكذا قمنا بإلغاء النمط Flashback.

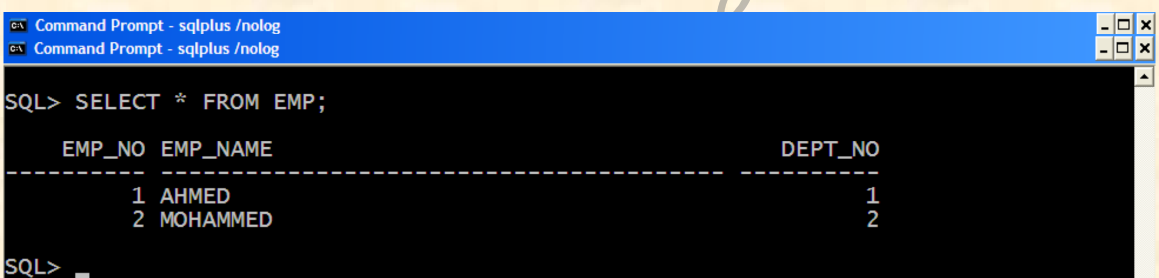
:B- Flashback Table

ويسمى ايضاً Flashback Table Query واستحدث هذا النوع فى الإصدار Oracle10g ويعتمد اساساً على الUndo Tablespace ، وهو الرجوع بالجدول الى فترة زمنية فى الماضى ويعتمد اساساً الى الذهاب الى الUndo Segments وارجاع الجدول الى لحظة معينة فى الماضى مستفيدا من المعلومات الموجودة Undo Segments ، فقد تكون جرت على الجدول عدد من التعديلات سنجدها فى الUndo Tablespace ، بالطبع قد تفشل عملية ال Flashback Table إذا لم نجد المعلومات المطلوبة فى الUndo Tablespace نتيجة لإنتهاء فترة الإحتفاظ بالمعلومات المطلوبة مثلاً.

واليك الان السناريو التالى :
لنفترض ان لدينا جولين الاول يسمى DEPT وهو جدول الإدارات والثانى يسمى EMP وهو جدول الموظفين.

فلنستعرض كلا الجدولين.

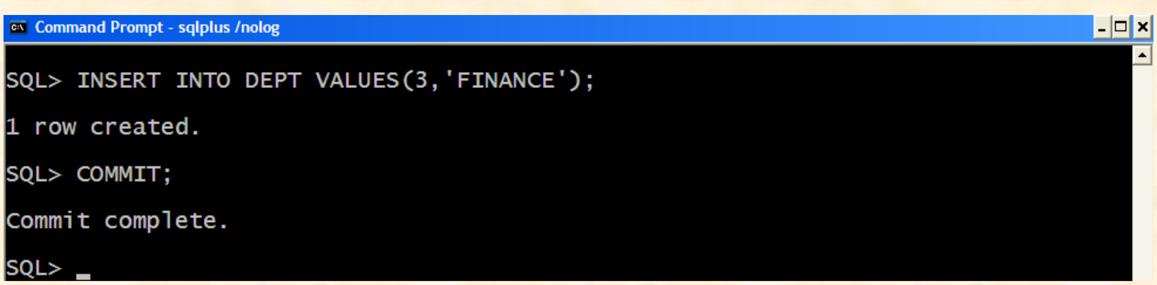
```
SELECT * FROM DEPT;
```




```
SELECT * FROM EMP;
```

لنضف إدارة جديد فى جدول الإدارات DEPT.

```
INSERT INTO DEPT VALUES(3,'FINANCE');
```



ونقوم بإضافة موظف جديد فى جدول الموظفين EMP بحيث ينتمى الموظف الجديد للإدارة الجديدة.

```
INSERT INTO EMP VALUES(3,'KHALED',3);
```

```
Select Command Prompt - sqlplus /nolog
SQL> INSERT INTO EMP VALUES(3,'KHALED',3);

1 row created.

SQL> COMMIT;

Commit complete.

SQL>
```

لنقوم بعرض الزمن الان

```
SELECT SYSDATE FROM DUAL;
```

```
Command Prompt - sqlplus /nolog
SQL> select sysdate from dual;

SYSDATE
-------------------
13-06-2008:01-01-06

SQL>
```

نحتاجة لعمل الFlashback Table.

الان نقوم بحذف الحقلين الذّين اضفناهما للجدولين.

```
DELETE EMP WHERE EMP_NO=3;
DELETE DEPT WHERE DEPT_NO=3;
```

```
Command Prompt - sqlplus /nolog
SQL> DELETE EMP WHERE EMP_NO=3;

1 row deleted.

SQL> DELETE DEPT WHERE DEPT_NO=3;

1 row deleted.

SQL> COMMIT;

Commit complete.

SQL> _
```

الان لنقوم بعمل Flashback Table للجدول EMP إلى ما قبل حذف الحقل EMP_NO=3 ،

```
FLASHBACK TABLE EMP TO TIMESTAMP
TO_TIMESTAMP('13-06-2008:01-01-06','DD-MM-
YYYY:HH24-MI-SS');
```

```
Command Prompt - sqlplus /nolog

SQL> FLASHBACK TABLE EMP TO TIMESTAMP TO_TIMESTAMP('13-06-2008:01-01-06','DD-MM-
YYYY:HH24-MI-SS');
FLASHBACK TABLE EMP TO TIMESTAMP TO_TIMESTAMP('13-06-2008:01-01-06','DD-MM-YYYY:
HH24-MI-SS')
                *
ERROR at line 1:
ORA-08189: cannot flashback the table because row movement is not enabled


SQL>
```

فشلت عملية الإسترجاع Flashback Table وذلك لأننا نحتاج لعمل Enable Row Movement ، وهو عبار عن Flag يتم وضعه فى الData Dictionary ليوضح للاوركل عمليات التعديل.

```
Command Prompt - sqlplus /nolog

SQL> ALTER TABLE EMP ENABLE ROW MOVEMENT;

Table altered.

SQL>
```

قمنا بعمل Enable Row Movement للجدول EMP ، فلنحاول الان عمل Flashback Table للجدول EMP مره اخرى.

```
Command Prompt - sqlplus /nolog

SQL> FLASHBACK TABLE EMP TO TIMESTAMP TO_TIMESTAMP('13-06-2008:01-01-06','DD-MM-
YYYY:HH24-MI-SS');
FLASHBACK TABLE EMP TO TIMESTAMP TO_TIMESTAMP('13-06-2008:01-01-06','DD-MM-YYYY:
HH24-MI-SS')
*
ERROR at line 1:
ORA-02091: transaction rolled back
ORA-02291: integrity constraint (TEST.FK_DEPT_NO) violated - parent key not
found


SQL>
```

رسالة خطأ اخرى ، هل عرفت لماذا ؟ لأنه قد تم إنتهاك القيد Foreign Key Constraint إذ لا يمكن إعادة الموظف رقم 3 الى الجدول EMP وهو ينتمى الى الإدارة تم حذفها ‘ فهو ينتمى للإدارة رقم 3 قد تم حذفها ، فما الحل إذاً؟

الحل ان نقوم بعمل Flashback Table لكلا الجدولين EMP & DEPT.
لكن لا تنسى أن تقوم بعمل Enable Row Movement.

```
FLASHBACK TABLE EMP,DEPT TO TIMESTAMP
TO_TIMESTAMP('13-06-2008:01-01-06','DD-MM-
YYYY:HH24-MI-SS');
```

```
Command Prompt - sqlplus /nolog

SQL> ALTER TABLE DEPT ENABLE ROW MOVEMENT;

Table altered.

SQL> FLASHBACK TABLE EMP,DEPT TO TIMESTAMP TO_TIMESTAMP('13-06-2008:01-01-06','D
D-MM-YYYY:HH24-MI-SS');

Flashback complete.

SQL>
```

لقد تمت عملية الFlashback Tables بنجاح وليس ضرورياً أن نقوم بترتيب الجدول فى عبارة الFlashback وذلك لأن القيود Constraint يتم التأكد منها بعد نهاية العملية Transaction.

ملاحظة : لو فشلت محاولة استرجاع جدول Flashback Table فى اى عملية فإن العملية تتوقف ومن ثم يتم التراجع ROLLBACK عن العملية كلها ، اى لو نجحت عملية استرجاع الجدول DEPT فى العملية السابقة ولكن فشلت عملية استرجاع الجدول EMP فإن العملية برمتها ستفشل ويتم التراجع عن العملية كلها.

كما لا حظت عملية الFlashback Table لسيت مضمونة النجاح ، فقد تفشل فى احد الاسبابا التالية :-
1- إذا تم انتهاك احد القيود Constraint Violated.
2- إذا لم يتم عمل Enable Row Movement للجدول المطلوب استرجاعه.
3- إذا لم تتوفر المعلومات المطلوبة فى ال Undo Tablespace ، ORA-08180:
"no snapshot found based on specified time
4- لا يمكن عمل Flashback Table لجدول فى Sys Schema.

C- Flashback Versions Query:

قد تحتاج احياناً كمدير لقاعدة بيانات لمراقبة بعض التعديلات على الحقول فتحتاج لمعرفة القيم التى اخذها حقل معين لفترات تاريخية معينة ، فمثلا قد تتحول مرتبات الموظفين كل فترة فيتم تعديل ملف الSalary كل فترة زمنية ، ولكن لنفترض أننى احتاج لمعرفة الHistory او القيم السابقة للمرتب لموظف معين منذ أن تم تعينة ، فكل ما هو متاح لى فى الجدول الان هو اخر قيمة للمرتب الحالي.

وقد استحدث هذا النوع فى الإصدار Oracle10g ويعتمد اساساً على ال Undo Tablespace. بحيث ياتى بالمعلومات التى يريدها من الUndo Segments بعد أن نحدد له الفترة الزمنية المحددة أو عن طريق System Change Number (SCN).

```
SELECT SALARY,VERSIONS_XID AS
XID,VERSIONS_STARTSCN,VERSIONS_ENDSCN,VERSIONS_O
PERATION FROM SALARY VERSIONS BETWEEN SCN
MINVALUE AND MAXVALUE  WHERE EMP_NO=1;
```

```
Command Prompt - sqlplus /nolog

SQL> SELECT SALARY,VERSIONS_XID AS XID,VERSIONS_STARTSCN,VERSIONS_ENDSCN,VERSION
S_OPERATION FROM SALARY VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE  WHERE EMP_NO
=1;

    SALARY XID              VERSIONS_STARTSCN VERSIONS_ENDSCN V
---------- ---------------- ----------------- --------------- -
      2000 06000800A2050000           2930015                 U
      1500 06000900A2050000           2930009         2930015 U
      1000 06002F00A1050000           2929973         2930009 I

SQL> _
```

لاحظ معى قيمة المرتب ابتداءً كانت 1000 وكانت نتيجة عمل Insert كما هو موضح بالحرف I ، بعد ذلك تم تعديلها الى القيمة 1500 نتيجة عمل Update كما هو موضح بالحرف U، وايضاً تم تعديله مرة اخرى الى القيمة 2000 بواسطة الامر Update كما هو موضح بالحرف U.

لقد قمنا فى المثال السابق بعرض المعلومات مستخدمين الSCN ، كما يمكن إستخدام الTimestamp .

.

ملاحظـــــــة: لا يتم عرض القيم التى لم يتم تثبيتها Commited.

إذاً عملية الFlashback Versions إذا لم تجد العلومات المطلوبة فى الUndo Tablespace فأنها ستفشل ، كما لا يمكن عمل Flashback Versions للجداول الخارجية External Tables وكذلك الجداول المؤقتة Temporary Tables وأيضاً المناظير Views.

D- Flashback Transaction:

فى كل من Flashback Table and Flashback versions يتم استخدام الUndo Data للكائنات ، أما الFlashback Transaction فإنه يقوم بإسترجاع جميع الUndo Data For transaction اى على مستوى العمليات مهما كانت العمليات مرتبطة بعدد من الكائنات ، للإستعلام عن ال Flashback Transaction نستخدم الView التالى : FLASHBACK_TRANSACTION_QUERY.

```
DESC FLASHBACK_TRANSACTION_QUERY;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> DESC FLASHBACK_TRANSACTION_QUERY;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------

 XID                                                RAW(8)
 START_SCN                                          NUMBER
 START_TIMESTAMP                                    DATE
 COMMIT_SCN                                         NUMBER
 COMMIT_TIMESTAMP                                   DATE
 LOGON_USER                                         VARCHAR2(30)
 UNDO_CHANGE#                                       NUMBER
 OPERATION                                          VARCHAR2(32)
 TABLE_NAME                                         VARCHAR2(256)
 TABLE_OWNER                                        VARCHAR2(32)
 ROW_ID                                             VARCHAR2(19)
 UNDO_SQL                                           VARCHAR2(4000)

SQL>
```

ولأن البيانات الموجودة فى هذا الView حساسة جداً لذلك فإن هذا ال View محمى بواسطة الصلاحية SELECT ANY TRANSACTION PRIVILEGE.

ولأن استعلام جميع العمليات على قاعدة البيانات يأتى بعدد كبير جداً من المعلومات ، فمن الأفضل التركيز فى الإستعلام بواسطة الشروط ، فلو أنى مثلاً أريد استعراض العمليات التى تمت على جدول معين فمن الافضل تحديد الشرط WHERE TABLE_NAME='TABLE_NAME'.

ولنفترض الان أنى اريد عمل استعلام عن العمليات التى حدثت على الجدول SALARY.

```
SELECT START_TIMESTAMP,UNDO_SQL,TABLE_NAME FROM
FLASHBACK_TRANSACTION_QUERY WHERE
TABLE_NAME='DEPT';
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> SELECT START_TIMESTAMP,UNDO_SQL,TABLE_NAME FROM  FLASHBACK_TRANSACTION_QUER
Y WHERE TABLE_NAME='DEPT';

START_TIM
---------
UNDO_SQL
--------------------------------------------------------------------------------

TABLE_NAME
--------------------------------------------------------------------------------

15-JUN-08
update "TEST"."DEPT" set "DEPT_NAME" = 'FINANCE' where ROWID = 'AAAMREAAEAAAAB1A

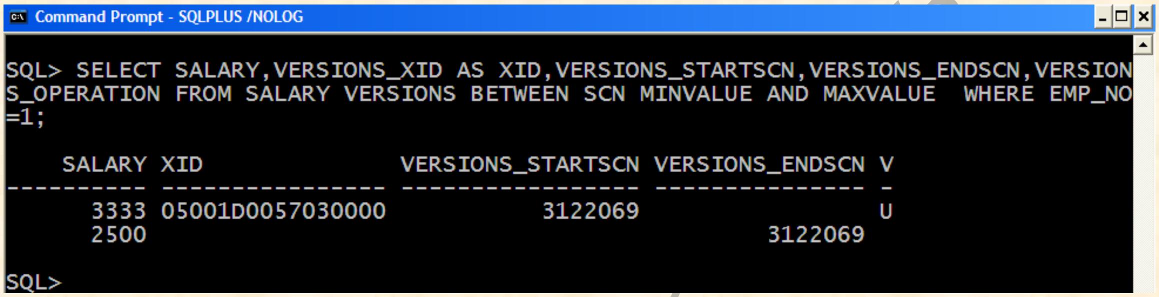
AC';
DEPT


SQL>
```

لقد تمت عملية UPDATE على الجدول DEPT.

ولكن يمكن استخدام الFLASHBACK VERSIONS اولاً لتحديد XID (Transaction Identifier)

```
SELECT SALARY,VERSIONS_XID AS
XID,VERSIONS_STARTSCN,VERSIONS_ENDSCN,VERSIONS_O
PERATION FROM SALARY VERSIONS BETWEEN SCN
MINVALUE AND MAXVALUE  WHERE EMP_NO=1;
```



```
Command Prompt - SQLPLUS /NOLOG

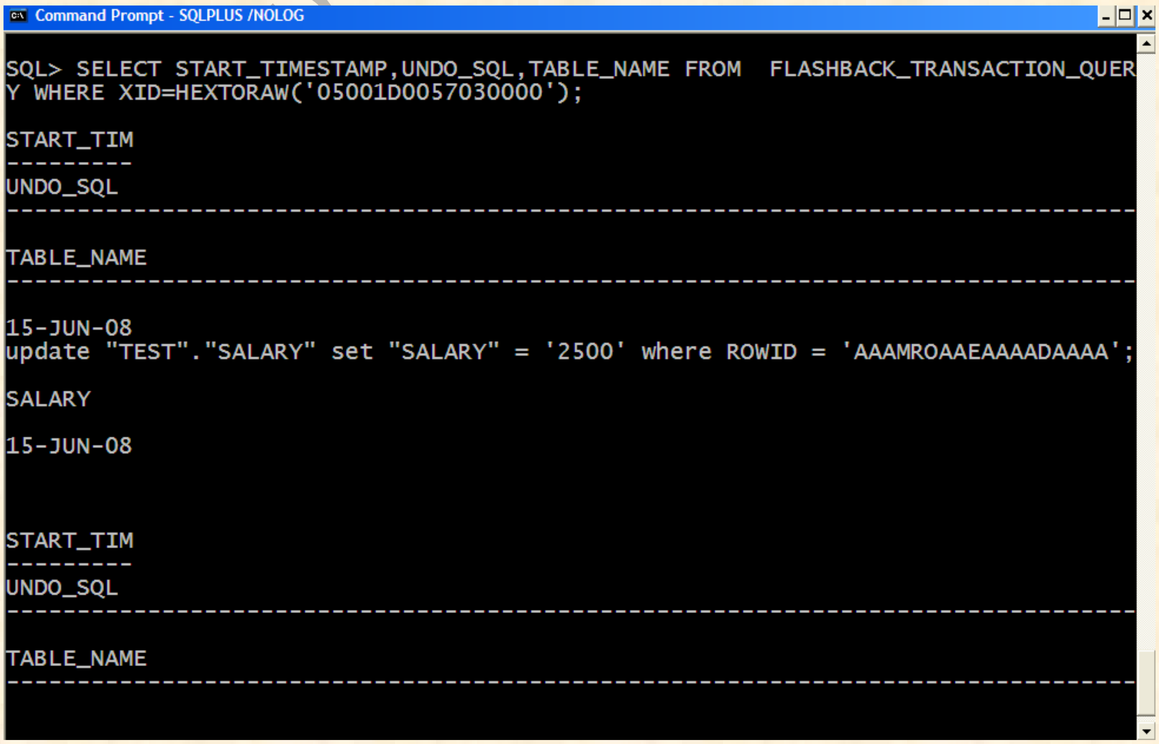
SQL> SELECT SALARY,VERSIONS_XID AS XID,VERSIONS_STARTSCN,VERSIONS_ENDSCN,VERSION
S_OPERATION FROM SALARY VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE  WHERE EMP_NO
=1;

    SALARY XID              VERSIONS_STARTSCN VERSIONS_ENDSCN V
---------- ---------------- ----------------- --------------- -
      3333 05001D0057030000           3122069                 U
      2500                                            3122069

SQL>
```

لاحظ معى XID=05001D0057030000 ، سوف نستخدمها فى ال Flashback Transaction Query.

```
SELECT START_TIMESTAMP,UNDO_SQL,TABLE_NAME FROM
FLASHBACK_TRANSACTION_QUERY WHERE
XID=HEXTORAW('05001D0057030000');
```



```
Command Prompt - SQLPLUS /NOLOG

SQL> SELECT START_TIMESTAMP,UNDO_SQL,TABLE_NAME FROM  FLASHBACK_TRANSACTION_QUER
Y WHERE XID=HEXTORAW('05001D0057030000');

START_TIM
---------
UNDO_SQL
--------------------------------------------------------------------------------

TABLE_NAME
--------------------------------------------------------------------------------

15-JUN-08
update "TEST"."SALARY" set "SALARY" = '2500' where ROWID = 'AAAMROAAEAAAADAAAA';

SALARY

15-JUN-08


START_TIM
---------
UNDO_SQL
--------------------------------------------------------------------------------

TABLE_NAME
--------------------------------------------------------------------------------
```

هكذا استخدمنا FLASHBACK_VERSIONS AND FLASHBACK_TRANSACTION.

وعموماً فإن الFLASHBACK_TRANSACTION يستخدم لعرض العمليات التى حدثت فى قاعدة البيانات ويستخدم هذا النوع ايضاً ال Undo Information ، ويسمى ايضاً FLASHBACK_TRANSACTION_QUERY.

:E- Flashback Drop

فى الإصدارات السابقة من قاعدة البيانات اوركل Oracle Database لحظة عمل حذف للجدول Drop Table فإنه سوف يزال ايضاً من الData Dictionary وسوف نقوم بعمل استرجاع Recovery لقاعدة البيانات إذا أردت أن استرجع جدول واحداً تم حذفه خطأً ، لا شك أنه حل مكلف وملكلف جداً من حيث الزمن ومن حيث أننا سأفقد بعض العمليات.

فعند حذف جدول فى الإصدارات السابقة من اوركل يتم تحرير المساحات التى كان يستقلها ، أما فى الإصدار Oracle10g لا يتم تحرير المساحات التى يستقلها الجدول وما يتبعه من كائنات لحظة حذف الجدول ، وإنما يتم وضع الجدول مؤقتاً فى سلة المهملات RECYCLE BIN ويظل الجدول مملوكاً ايضاً للمستخدم رغم إنتقاله الى سلة المهملات ، ولكن بالطبع لحظة إنتقاله إلى سلة المهملات يتم إعادة تسميته حتى لا يحدث تضارب فى الاسماء إذا أردت أن اقوم بإنشاء جدول اخر بنفس الإسم ، هذا الوضع يتيح لنا فرصة استرجاع الجدول من سلة المهملات دون الحوجة لاسترجاع قاعدة البيانات ، مما يقلل لنا فترة الاسترجاع وكذلك دون أن نفقد اى عمليات .

بكل بساطة سلة المهملات RECYCLE BIN فى قاعدة البيانات اوركل تشبه الى حد كبير سلة المهلات فى نظام التشغيل ويندوز WINDOWS.

يمكن الإستعلام عن سلة المهملات RECYCLE BIN بواسطة:

DBA_RECYCLEBIN

USER_RECYCLEBIN

كما يمكن الإستعلام عن طريق الامر:

SHOW RECYCLEBIN

والان لنتتابع معاً هذا السناريو:
يقوم المستخدم بحذف الجدول USER_MASTER ، ومن ثم نقوم بإستلاعم عن سلة المهملات RECYCLE BIN.

```
DROP TABLE USER_MASTER;
```

```
SHOW RECYCLE BIN;
```

```
Command Prompt - SQLPLUS /NOLOG
SQL> SELECT * FROM TAB;

TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
USER_MASTER                    TABLE

SQL> DROP TABLE USER_MASTER;

Table dropped.

SQL> SELECT * FROM TAB;

TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
BIN$XgyOK9KFQVqqv0+7CfAtuQ==$0 TABLE

SQL> SHOW RECYCLEBIN;
ORIGINAL NAME    RECYCLEBIN NAME                OBJECT TYPE  DROP TIME
---------------- ------------------------------ ------------ -------------------

USER_MASTER      BIN$XgyOK9KFQVqqv0+7CfAtuQ==$0 TABLE        2008-06-15:12:53:57

SQL>
```

لاحظت معي كيف أنه رغم أننا قمنا بحذف الجدول USER_MASTER إلا أن ما زال موجوداً فى سلة المهملات ويمكننا كذلك استرجاعه .

```
FLASHBACK TABLE USER_MASTER TO BEFORE DROP;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> SELECT * FROM TAB;

TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
BIN$XgyOK9KFQVqqv0+7CfAtuQ==$0 TABLE

SQL> FLASHBACK TABLE USER_MASTER TO BEFORE DROP;

Flashback complete.

SQL> SELECT * FROM TAB;

TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
USER_MASTER                    TABLE

SQL>
```

لقد قمنا بإسترجاع الجدول من سلة المحذوفات لكن قد تفشل عملية الاسترجاع إذا قمنا بإنشاء جدول بنفس

الاسم قبل عملية الإسترجاع لذلك يمكن اعادة تسميته لتفادى عملية تضارب الاسماء.

```
FLASHBACK TABLE USER_MASTER TO BEFORE DROP
RENAME TO MASTER:
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> DROP TABLE USER_MASTER;

Table dropped.

SQL> FLASHBACK TABLE USER_MASTER TO BEFORE DROP RENAME TO MASTER;

Flashback complete.

SQL> SELECT * FROM TAB;

TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
MASTER                         TABLE

SQL> _
```

لقد قمنا بإسترجاع الجدول ولكن بإسم جديد .

```
SELECT * FROM USER_RECYCLEBIN;
```

```
Command Prompt - SQLPLUS /NOLOG

SQL> SELECT * FROM USER_RECYCLEBIN;

no rows selected

SQL> _
```

لا يوجد جدول فى سلة المهملات .

يمكن تنظيف سلة المهملات ومسح الجداول من سلة المهملات بواسطة الامر PURGE.

```
Command Prompt - SQLPLUS /NOLOG

SQL> DROP TABLE MASTER;

Table dropped.

SQL> SHOW RECYCLEBIN;
ORIGINAL NAME    RECYCLEBIN NAME                OBJECT TYPE  DROP TIME
---------------- ------------------------------ ------------ -------------------
MASTER           BIN$Wy2BpegDRwuYKUCBbKTEdg==$0 TABLE        2008-06-15:13:29:56

SQL> PURGE TABLE MASTER;

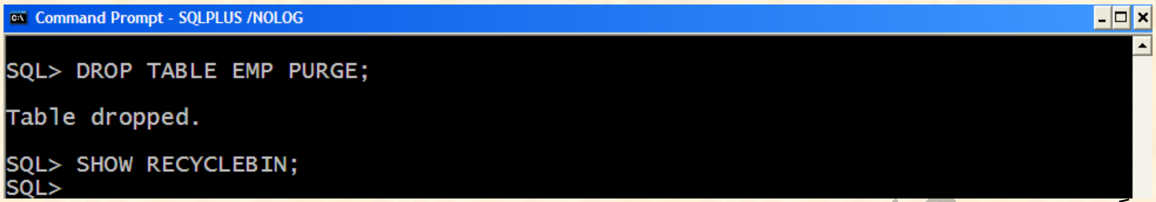
Table purged.

SQL> SHOW RECYCLEBIN;
SQL>
```

لا يوجد الان الجدول MASTER فى سلة المهملات .

كذلك يمكن حذف الجدول عن طريق المستخدم دون وضع الجدول فى سلة المهملات وانما حذفه مباشرة من قاعدة البيانات بواسطة الامر DROP TABLE <TABLE_NAME> PURGE

```
DROP TABLE EMP PURGE;
```



إذاً من الحالات التى يتم فيها حذف الجداول من قاعدة البيانات دون المرور بسلة المهملات :-

1- استخدام الامر DROP TABLE <TABLE_NAME> PURGE.

2- لحظة حذف الTABLESPACE الذى يحوى الجداول بواسطة الامر DROP TABLESPACE <TABLESPACE_NAME> INCLUDING CONTENTS.

3- عند حذف المستخدم الذى يحوى الجداول بواسطة الامر DROP USER <USER_NAME> CASCADE.

F- Flashback Database:

هل فكرت يوماً ماذا تفعل لو قمت بحذف مستخدمٍ ما خطأً ؟ الجواب وبكل بساطة اقوم بعمل استرجاع لقاعدة البيانات Recovery من النسخ الاحتياطى Backup.

هذا الحل صحيح ولكن مكلف من حيث الزمن خصوصاً إذا كانت قاعدة البيانات كبيرة ، فإسترجاع احجام كبيرة من الملفات لا بد أنه مكلف من حيث الزمن .

ما الحل إذاً ، الحل هو Flashback Database.

الFlashback Database هى إرجاع قاعدة البيانات الى نقطة زمنية معينة فى الماضى ، بحيث جميع التعديلات التى حدثت فى قاعدة البيانات من تلك النقطة الى الان كأن لم تكن.

نستفيد من الFlashback Database فى حالة حدوث اخطاء منطقية ، كأن يحذف مدير قاعدة البيانات مستخدم عن طريق الخطأ أو عمل Truncate Table ، أما الأخطاء الفيزيائية كأن نفقد Data File او غيره من الملفات فلا نستطيع الإستفادة من الFlashback Database وكل ما نستطيعه هو عمل استرجاع للملفات من النسخ الإحتياطى بالطريقة التقلدية.

عملية الFlashback Database هى عملية سريعة جداً للرجوع بقاعدة البيانات لفترة زمنية فى الماضى مقارنة بعملية الإسترجاع من ملفات النسخ الإحتياطى بالطريقة التقلدية وذلك لأننا لا نحتاج ارجاع ملفات النسخ الإحتياطى لقاعدة البيانات.

لحظة عمل Enable Flashback Database يتم إنشاء Background Process يسمى Recovery Writer (RVWR) ويتم تخصيص جزء فى الذاكرة SGA يسمى Flashback Buffer ويتم تخصيص جزء من الدسيك لل Flashback logs وهذا الجزء هو دائماً الFlash Recovery Area .
يكون السناريو بحيث يتم وضع كل Block تم تعديله فى قاعدة البيانات الى الFlash Buffer بعد ذلك يقوم الRVWR بوضع هذه الBlocks من ال Flashback Buffer إلى الFlashback Logs .

هذا السناريو يشبه الى حد كبير سناريو الRedo Log Buffer والRedo Log Files ولكن الإختلاف فى أنه ليس كل التعديلات تكتب فى الFlashback Buffer كما يحدث فى الRedo Log Buffer وإنما Complete Block Images ، كذلك الFlashback Loges لا تتم ارشفته كما يحدث للRedo Log Files.

كما ذكرت أنه ليست كل التعديلات تكتب فى الFlashback Buffer وإنما Complete Block Images ، لذا فإن هناك كثير من التعديلات تحدث ولا يتم كتابتها مباشرة فى الFlashback Buffer وإنما يتم تأخيرها وربما تحدثت تعديلات اخرى قبل أن تكتب فى ال Flashback Buffer ، ما أرمى اليه أنا أنه قد يتم كتابة Complete Block وهو يحوى مجموعة مختلفة من التعديلات حدثت فى ازمنة مختلفة ، لذا عملية الFlashback Database لوقت مضبوط هى من الصعوبة بمكان ، لذا فإن الFlashback Database يحتاج لأن تكون قاعدة البيانات فى النمط Archive Log Mode وذلك لأن الارشيف يحتوى على جميع تفاصيل التعديلات التى حدثت فى قاعدة البيانات على خلاف Flashback Logs ، إذا الFlashback Logs مع الArchive Logs يستطيعان أن ينجحا عملية ال Flashback Database

لتهيئة الFlashback Database نتبع الخطوات الاتية:-

1- التأكد من أن قاعدة البيانات تعمل فى النمط Archive Log Mode.

تحدثنا فى فصول سابقة على أن قاعدة البيانات تعمل على الاقل على اثنين من Redo Log Files وهى ملفات تحوى على التعديلات التى تمت فى قاعدة البيانات .
وأن LGWR Background Process يقوم بكتابة التعديلات الموجودة فى الRedo Log Buffer ويضعها فى الRedo Log Files ، وتكون عملية الكتابة بشكل دورى بين الRedo Log ويتم إعادة الكتابة فى الRedo Logs ،

مما يؤدى لفقدان البيانات الموجودة فيه لذا كان الحل هو تشغيل قاعدة البيانات فى الانمط Archive Log Mode بحيث يتم إنشاء Background Process جديد يسمى ARCn بحيث يقوم بعمل نسخ لملفات الRedo Log Files بحيث لا نفقد البيانات الموجودة فى هذه الملفات والتى نعتمد عليها فى عملية الإسترجاع Recovery ؛ والتى بدونها تفشل عملية الRecovery.

لمعرفة هل قاعدة البيانات تعمل فى الوضع Archive Log Mode ام لا؟

```
SELECT LOG_MODE FROM V$DATABASE;
```

```
Command Prompt - sqlplus /nolog

SQL> SELECT LOG_MODE FROM V$DATABASE;

LOG_MODE
------------
NOARCHIVELOG

SQL> _
```

إذا قاعدة البيانات فى الوضع No Archive Log.

لتهيئة قاعدة البيانات فى الوضع Archive Log نقوم بالخطوات الاتية:

A- تحديد المكان الذى سوف نضع فيه الأرشيف بواسطة المتغير LOG_ARCHIVE_DEST مع مراعاة أنه يمكن تحديد 10 إتجاهات مختلفة وذلك بواسطة المتغيرات LOG_ARCHIVE_DEST_n حيث n تاخذ القيم من 1 الى 10.

```
ALTER SYSTEM SET
LOG_ARCHIVE_DEST_1='LOCATION=D:\ARCHIVE\’
SCOPE=SPFILE;
```

```
Command Prompt - sqlplus /nolog
Command Prompt - SQLPLUS /NOLOG

SQL> alter system set log_archive_dest_1='LOCATION=D:\archive\' scope=spfile;

System altered.
```

كما ذكرت سابقاً يمكن تحديد اتجاهات اخرى للارشيف ، كما يمكن تهيئة متغيرات اخرى لكنها غير ضرورية مثلاً LOG_ARCHIVE_FORMAT هذا المتغير لتحديد الفورمات التى يخرج بها ملف الارشيف ، فى الاصل يأخذ القيمة .

حيث:-

Variable	Description
%d	A unique database identifier, necessary if multiple databases are being archived to the same directories.
%t	The thread number, visible as the THREAD# column in V$INSTANCE. This is not significant, except in a RAC database.
%r	The incarnation number. This is important if an incomplete recovery has been done, as described in Chapter 27.
%s	The log switch sequence number. This will guarantee that the archives from any one database do not overwrite each other.

B- نغلق قاعدة البيانات ونفتحها فى الوضع Mount ، ثم ننفذ الامر ALTER DATABASE ARCHIVELOG.

```
ALTER DATABASE ARCHIVELOG;
```

Command Prompt - SQLPLUS /NOLOG

```
SQL> SHUT IMMEDIATE;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> STARTUP MOUNT;
ORACLE instance started.

Total System Global Area  171966464 bytes
Fixed Size                   787988 bytes
Variable Size             145488364 bytes
Database Buffers           25165824 bytes
Redo Buffers                 524288 bytes
Database mounted.
SQL> ALTER DATABASE ARCHIVELOG;

Database altered.

SQL> ALTER DATABASE OPEN;

Database altered.

SQL> SELECT LOG_MODE FROM V$DATABASE;

LOG_MODE
------------
ARCHIVELOG

SQL> _
```

الان قاعدة البيانات تعمل فى النمط ARCHIVELOG MODE .

يمكن الاستعلام ايضاً عن عمل ARCHIVELOG MODE بواسطة:

```
SELECT ARCHIVER FROM V$INSTANCE;
```

```
SQL> SELECT ARCHIVER FROM V$INSTANCE;

ARCHIVE
-------
STARTED

SQL>
```

لحظة عمل SWITCH LOGFILE يتم عمل نسخ لملف الREDO LOG FILE للارشيف.

```
ALTER SYSTEM SWITCH LOGFILE;
```

كما يمكن الإستعلام عن ملفات الارشيف

```
SELECT NAME FROM V$ARCHIVED_LOG;
```

```
SQL> ALTER SYSTEM SWITCH LOGFILE;

System altered.

SQL> SELECT NAME FROM V$ARCHIVED_LOG;

NAME
--------------------------------------------------------------------------------

D:\ARCHIVE\ARC00257_0654318341.001
D:\ARCHIVE\ARC00258_0654318341.001

SQL>
```

هذه هى ملفات الارشيف.

2- تهيئة الFlash Recovery Area:

ذكرت سابقاً أن الFlashback Logs لا تتم إدارته إلى فى الFlash Recovery Area وهى مكان يتم تخصيصة بواسطة المتغير DB_RECOVERY_FILE_DEST ويتم تحديد مساحة هذا المكان المخصص بواسطة هذا المتغير DB_RECOVERY_FILE_DEST_SIZE ، إذاً لتهيئة ال Flash Recovery Area يلزمنا تهيئة المتغيران (DB_RECOVERY_FILE_DEST & DB_RECOVERY_FILE_DEST_SIZE)

```
ALTER SYSTEM SET DB_RECOVERY_FILE_DEST='D:\
ORACLE\PRODUCT\10.1.0\FLASH_RECOVERY_AREA;
```

```
SQL> ALTER SYSTEM SET DB_RECOVERY_FILE_DEST='D:\oracle\product\10.1.0\flash_reco
very_area';

System altered.
```

```
ALTER SYSTEM SET DB_RECOVERY_FILE_DEST_SIZE=2G;
```

```
SQL> _
```

3- تحديد فترة الإحتفاظ بال Flashback Log:

وذلك قبل أن يتم استخدامه وإعادة الكتابة فيه مرة اخرى ويتم تحديد ذلك

```
SHOW PARAMETER DB_RECOVERY_FILE_DEST;
```

```
SQL> SHOW PARAMETER DB_RECOVERY_FILE_DEST;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
db_recovery_file_dest                string      D:\oracle\product\10.1.0\flash
                                                 _recovery_area
db_recovery_file_dest_size           big integer 2G
SQL>
```

بواسطة المتغير DB_FLASHBACK_RETENTION_TARGET وهو زمن الاحتفاظ بالدقائق والاصل أن يتم الإحتفاظ لمدة يوم.

4- اغلاق وفتح قاعدة البيانات فى الوضع Mount:

```
SQL> ALTER SYSTEM SET DB_FLASHBACK_RETENTION_TARGET=1440;

System altered.

SQL> _
```

ومن ثم تمكين الFlashback بواسطة الامر ALTER DATABASE FLASHBACK ON

```
ALTER DATABASE FLASHBACK ON;
```

```
Command Prompt - sqlplus /nolog

SQL> SHUT IMMEDIATE;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> STARTUP MOUNT;
ORACLE instance started.

Total System Global Area  171966464 bytes
Fixed Size                   787988 bytes
Variable Size             145488364 bytes
Database Buffers           25165824 bytes
Redo Buffers                 524288 bytes
Database mounted.
SQL> ALTER DATABASE FLASHBACK ON;

Database altered.

SQL> ALTER DATABASE OPEN;

Database altered.

SQL>
```

هكذا انتهينا من تهيئة الFlashback.

للتأكد من تهيئة الFlashback نقوم بعمل الاستعلام الاتى:

```
SELECT FLASHBACK_ON FROM V$DATABASE;
```

```
Command Prompt - sqlplus /nolog

SQL> SELECT FLASHBACK_ON FROM V$DATABASE;

FLA
---
YES

SQL> _
```

لمراقبة الFlashback

```
SELECT * FROM V$FLASHBACK_DATABASE_LOG;
```

```
Command Prompt - sqlplus /nolog

SQL> SELECT * FROM V$FLASHBACK_DATABASE_LOG;

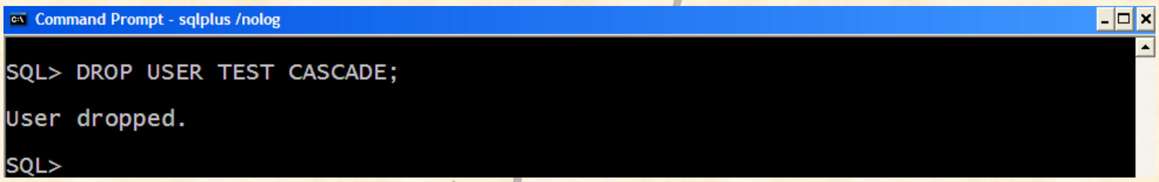
OLDEST_FLASHBACK_SCN OLDEST_FL RETENTION_TARGET FLASHBACK_SIZE
-------------------- --------- ---------------- --------------
ESTIMATED_FLASHBACK_SIZE
------------------------
             3434199 17-JUN-08             1440        8192000
                       0

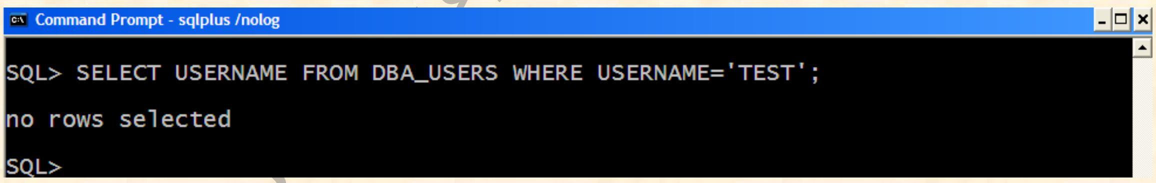
SQL>
```

لمعرفة كيفية استخدام الFlashback Database نتابع معاً هذا السناريو:

1- مدير قاعدة البيانات قام بحذف المستخدم TEST عن طريق الخطأ.

```
DROP USER TEST CASCADE;
```

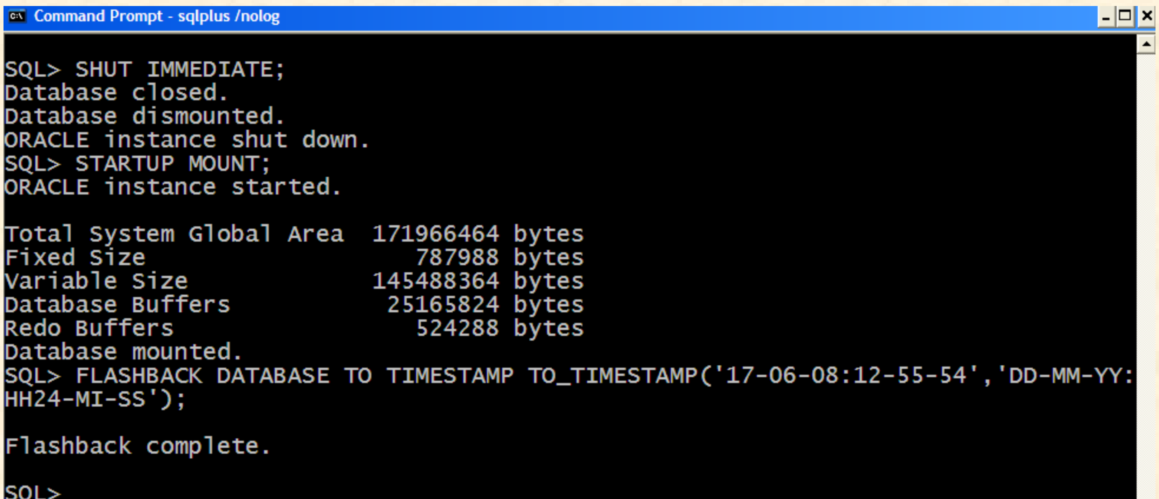




الان لا يوجد مستخدم اسمه TEST فى قاعدة البيانات.

2- اغلاق قاعدة البيانات وفتحها فى الوضع Mount ومن ثم عمل Flashback Database للزمن 17_06-08:12-55-54 وهو الوقت ما قبل حذف المستخدم TEST.

```
FLASHBACK DATABASE TO TIMESTAMP
TO_TIMESTAMP('17-06-08:12-55-54','DD-MM-YY:HH24-
MI-SS');
```



```
Command Prompt - sqlplus /nolog

SQL> SHUT IMMEDIATE;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> STARTUP MOUNT;
ORACLE instance started.

Total System Global Area  171966464 bytes
Fixed Size                   787988 bytes
Variable Size             145488364 bytes
Database Buffers           25165824 bytes
Redo Buffers                 524288 bytes
Database mounted.
SQL> FLASHBACK DATABASE TO TIMESTAMP TO_TIMESTAMP('17-06-08:12-55-54','DD-MM-YY:
HH24-MI-SS');

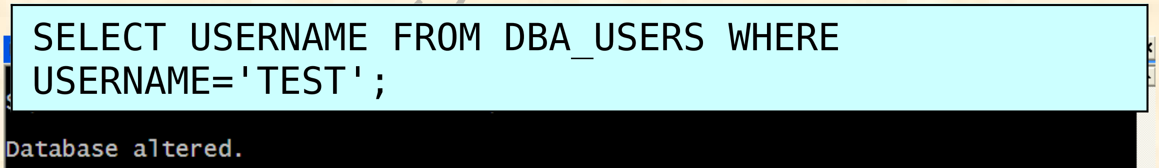
Flashback complete.

SQL>
```

3- نفتح قاعدة البيانات فى الوضع Resetlogs .

```
ALTER DATABASE OPEN RESETLOGS;
```

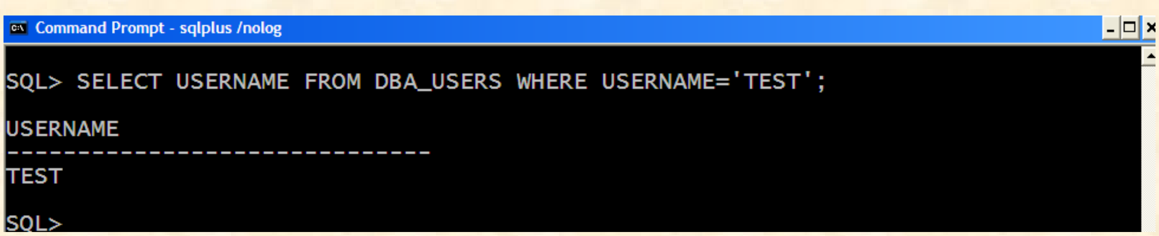
```
SELECT USERNAME FROM DBA_USERS WHERE
USERNAME='TEST';
```



```
Database altered.
```

4- اخيراَ لنتأكد من وجود المستخدم TEST.

أما إذا لم نكن نعرف مثلاً وقت حذف المستخدم المستخدم TEST فليس عندنا حل سواء التخمين ومن ثم فتح قاعدة البيانات فى الوضع Read Only والتأكد من جدوى الاسترجاع وقد نلجأ لإغلاق قاعدة البيانات وفتحها مرة اخرى



```
Command Prompt - sqlplus /nolog

SQL> SELECT USERNAME FROM DBA_USERS WHERE USERNAME='TEST';

USERNAME
------------------------------
TEST

SQL>
```

ومن ثم إعادة الFlashback Database لوقت اخر ، ولان نفتح قاعدة البيانات

مباشرة فى الوضع Resetlogs لان الخيار Restlogs يقوم بإعادة تهيئة الLog Sequence Number ويبدأ بأخذ القيمة (001) من جديد.